COMMUNICATION AND ALIGNMENT

OF GROUNDED SYMBOLIC KNOWLEDGE

AMONG HETEROGENEOUS ROBOTS

A Dissertation
Presented to
The Academic Faculty

by

Zsolt Kira

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in Computer Science in the
School of Interactive Computing
College of Computing

Georgia Institute of Technology
May 2010

COMMUNICATION AND ALIGNMENT
OF GROUNDED SYMBOLIC KNOWLEDGE
AMONG HETEROGENEOUS ROBOTS

Approved by:

Dr. Ronald C. Arkin, Advisor
School of Interactive Computing
College of Computing
Georgia Institute of Technology

Dr. Tucker Balch
School of Interactive Computing
College of Computing
Georgia Institute of Technology

Dr. Thomas R. Collins
Georgia Tech Research Institute

Dr. Ashok K. Goel
School of Interactive Computing
College of Computing
Georgia Institute of Technology

Dr. Charles L. Isbell, Jr.
School of Interactive Computing
College of Computing
Georgia Institute of Technology

Date Approved: April 5, 2010

for my wife, my family, and my friends,
whose support made this possible

ACKNOWLEDGEMENTS

First, I would like to thank my advisor, Dr. Ron Arkin, whose support has made this dissertation possible. His critical reading of this work continually challenged me to improve upon it, and I think the result is much improved and much more accessible as a result. His vast knowledge of the field has been extremely helpful during the exploration process that has led to this topic.

I would also like to thank the rest of the committee: Tucker Balch, Tom Collins, Ashok Goel, and Charles Isbell. The many helpful comments, especially during the formative stage of this work, helped me to better shape and define a substantive topic in robotics.

Without my family, this dissertation would have never been possible. My parents were inspirational in their ability to start from nothing and achieve success despite the challenges. They continually pushed me to work hard and excel at whatever I do, and without this push I would have never reached this stage. My brother, who always provided a positive role model that I could follow, was always supportive of my goals and was there for me when I needed help, no matter what the situation was. Thank you, family, for everything you have done for me.

I don't think I can ever thank my wife enough for her eternal patience, love, and support that she has shown throughout the process of writing this dissertation. Her sacrifices that allowed me to chase my dreams were great, and deeply appreciated. She never let me falter and constantly reminded me what I was working towards.

Special thanks go to Dr. Stephen Murrell, an incredible teacher and influential mentor while I was at the University of Miami. I would probably have not tread down this path had his Artificial Intelligence class not inspired me.

During my journey at Georgia Tech, I have met many amazing, intelligent, and fun-to-be-with friends and colleagues. They made the process much more enjoyable, and provided endless support, encouragement, and advice. To Patrick Ulam and Alan Wagner, thanks for the friendship from day one, filled with long discussions about robotics and life, lots of fun, and memorable trips. Going through the process in parallel with close friends, from start to finish, makes everything more enjoyable and helps to keep one's sanity. Endo Yoichiro was a great friend and constant leader in the lab, and his help was invaluable in learning the ropes. To Raffay Hamid and Maya Cakmak, thanks for the deep conversations, much-needed fun outside of school, and advice that always made sense. To Ana Fiallos, a friend for almost a decade, thanks for the invaluable friendship and for listening to all of my venting. I would also like to thank the many friends and colleagues that have, in some form or another, shaped my life at Georgia Tech: Ananth Rangathan, "Breakthrough" George Baah, Michael Kaess, Alex Stoytchev, Eric Martinson, Lilia Moshkina, Sung Hyun Park, and Mattia Castelnovi, all of whom provided great company at the Mobile Robot Lab; and Arya Irani, Soorajh Bhat, Keith O' Hara, and Matt Powers, for providing friendship and perspectives from outside the lab.

Finally, I greatly enjoyed all of my summer internships at the Navy Center for Applied Research in Artificial Intelligence at the Naval Research Laboratory. They broadened my perspective of the field and gave me invaluable experience. I would like to thank

Alan Schultz and Mitchell Potter for giving me the opportunity to be a part of this great laboratory, as well as the many friends and colleagues I met there: Don Sofge, Bob Daley, David Aha, Paul Wiegand, Ben Fransen, Brandon James, Bill Spears, Wende Frost, Frederick Heckel, Sam Blissard, Magda Bugaska, and Jeff Bassett.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

Experience forms the basis of learning. It is crucial in the development of human intelligence, and more broadly allows an agent to discover and learn about the world around it. Although experience is fundamental to learning, it is costly and time-consuming to obtain. In order to speed this process up, humans in particular have developed communication abilities so that ideas and knowledge can be shared without requiring first-hand experience.

Consider the same need for knowledge sharing among robots. Based on the recent growth of the field, it is reasonable to assume that in the near future there will be a collection of robots learning to perform tasks and gaining their own experiences in the world. In order to speed this learning up, it would be beneficial for the various robots to share their knowledge with each other. In most cases, however, the communication of knowledge among humans relies on the existence of similar sensory and motor capabilities. Robots, on the other hand, widely vary in perceptual and motor apparatus, ranging from simple light sensors to sophisticated laser and vision sensing.

This dissertation defines the problem of how heterogeneous robots with widely different capabilities can share experiences gained in the world in order to speed up learning. The work focus specifically on differences in sensing and perception, which can be used both for perceptual categorization tasks as well as determining actions based on environmental features. Motivating the problem, experiments first demonstrate that heterogeneity does indeed pose a problem during the transfer of object models from one

robot to another. This is true even when using state of the art object recognition algorithms that use SIFT features, designed to be unique and reproducible.

It is then shown that the abstraction of raw sensory data into intermediate categories for multiple object features (such as color, texture, shape, etc.), represented as Gaussian Mixture Models, can alleviate some of these issues and facilitate effective knowledge transfer. Object representation, heterogeneity, and knowledge transfer is framed within Gärdenfors' conceptual spaces, or geometric spaces that utilize similarity measures as the basis of categorization. This representation is used to model object properties (e.g. color or texture) and concepts (object categories and specific objects).

A framework is then proposed to allow heterogeneous robots to build models of their differences with respect to the intermediate representation using joint interaction in the environment. Confusion matrices are used to map property pairs between two heterogeneous robots, and an information-theoretic metric is proposed to model information loss when going from one robot's representation to another. We demonstrate that these metrics allow for cognizant failure, where the robots can ascertain if concepts can or cannot be shared, given their respective capabilities.

After this period of joint interaction, the learned models are used to facilitate communication and knowledge transfer in a manner that is sensitive to the robots' differences. It is shown that heterogeneous robots are able to learn accurate models of their similarities and difference, and to use these models to transfer learned concepts from one robot to another in order to bootstrap the learning of the receiving robot. In addition, several types of communication tasks are used in the experiments. For example, how can a robot communicate a distinguishing property of an object to help another robot differentiate it from its surroundings? Throughout the dissertation, the claims will be validated through both simulation and real-robot experiments.

CHAPTER 1

INTRODUCTION

"The only source of knowledge is experience" – Albert Einstein

"Experience: that most brutal of teachers. But you learn, my God do you learn." – C.S. Lewis

Experience forms the basis of learning. It is crucial in the development of human intelligence, and allows an agent to discover and learn about the world around it. Although experience is fundamental to learning, there is a finite amount of time one can gain it in. Each interaction requires performing an action that changes the world somehow, waiting for that action to affect the world, and determining the resulting effect. This can be extremely time-consuming, especially where the consequences of an action are delayed in time. Furthermore, in order to find patterns and learn, multiple interactions are required over time. Learning of perceptual categories via supervised training is also intensive, especially since it requires many instances. Human development and learning of new expertise, for example, takes many years to occur. In order to speed this process up, humans in particular have developed communication and artifacts that one can study without requiring first-hand experience. Furthermore, humans live in a social society where information is constantly exchanged through interaction, spoken language, and written communication. Many have even attributed such communication and socialization as contributing to human-level success (Donald, 1991).

Consider the same need for knowledge sharing among robots. Based on the recent growth of the field, it is reasonable to assume that in the near future there will be a collection of robots learning to perform tasks and gaining their own experiences in the world. Knowledge for the performance of such tasks can be programmed, but can also be taught by humans or learned autonomously. In order to speed up this learning, it will

Figure 1 – Variety of robots in use today.

be beneficial for the various robots to share their knowledge with each other. In most cases, however, the communication of knowledge among humans relies on the fact that similar sensory and motor capabilities as well as general cultural socialization are shared. Although variations exist and capabilities are affected by development, there is a large overlap. Robots, on the other hand, widely vary in perceptual and motor apparatus, ranging from a simple LEGO mindstorm robot with small wheels and primitive touch and light sensors, all the way to fully capable mobile manipulator robots with sophisticated ladar and vision sensing. There can be slight perceptual differences even among two robots of the same model. For example, the camera color characteristics may differ slightly. Figure 1 shows images of a small sample of the large variety of robots available today.

Different robots will also vary with respect to the types of sensory processing they perform, representations used, and level of experience and knowledge of the world they have. Unless a standard robotic platform is created in order to solve the wide variety of tasks for which robots are created, and the learning of such tasks is standardized, robots will likely have to bridge their differences before communicating and sharing experiences. This issue is especially important in emerging fields such as developmental robotics (Lungarella et al., 2003). Developmental robotics attempts to study robotics from the perspective of building capabilities progressively via embodied interaction with the world. In the single-robot case, exploration in the world is performed alone and can involve trying to find cause-effect rules (e.g. (Drescher, 1991)) or exploration of the robot's own capabilities (e.g. (Stoytchev, 2003)). Here, too, it is crucial that robots be able to share knowledge either through explicit communication or implicit means such as imitation. Such knowledge sharing speeds up development significantly and can allow more experienced robots to impart their wisdom to others. Social aspects of development have been recognized by developmental psychologists such as Vygotsky (Wertsch, 1995). However, the knowledge learned via such exploration of the world is embodiment-specific, that is unique to the particular sensing capabilities of the robot (Stoytchev, 2009). Reconciling the embodied nature of learning with the fact that the robots must also be able to communicate and share meanings is an important problem in this field. Hence, the need for the two robots to account for their differences before communication is important.

Note that while heterogeneity can present challenges, such as in the case of the transfer of learned knowledge, it can present opportunities as well. For example, the fusion of different types of information across multiple sensors on different robots can be extremely useful and can be leveraged to increase task performance. In other words, there is an upside to heterogeneity as well. This is why understanding and modeling heterogeneity when it does pose a challenge, the topic of this dissertation, is important.

This thesis defines and provides solutions to the problem of how heterogeneous robots with widely different capabilities can share learned knowledge gained in the world. The main motivation is to speed up learning, but important secondary capabilities such as the facilitation of cooperation in a joint task will also be demonstrated. We focus specifically on differences in sensing and perception, which can be used both for perceptual categorization tasks as well as determining world state in order to decide which action to perform.

1.1. Applications and Domains

Solving this problem would be useful in many application domains. Consider, for example, a robot similar to the Roomba that performs a vacuuming task in a household setting. Suppose that unlike the current version of the Roomba, the robot creates a map of the house using laser sensing and allows users to designate objects or areas that are off-limits via a symbolic representation (e.g. “couch” or “ball”). It would presumably take a large period of training for the robot to build this map, learn what objects look like, and learn the preferences and locations designated by its owner. Now suppose that the robot is upgraded to the latest version but this time it has a cheaper sonar sensor instead of a laser but also an additional camera. It would be inconvenient to burden the user with teaching the new robot skills, preferences, and limits that she has already taught to the older robot. As the number of such robotic devices increases, repeated hands-on instruction will become not just burdensome but impractical. If the newer robot could learn from the older robot’s experiences, this process would be greatly sped up. In addition, it would obviate or substantially reduce the need for user interaction. However, to share such knowledge they must first have a dialogue and explore the world in a targeted manner together in order to learn what their differences are.

Communication of knowledge among heterogeneous robots is not just useful when a single robot is replaced, but also in distributed multi-agent systems. Multiple robots will be working together with some shared goals, and communication between them is crucial. This might be useful in a military domain where robots are expected to be increasingly used. For example, how can a ground vehicle that has learned how to track a moving object share what it has learned with an aerial vehicle? Another example would be the addition or replacement of a new type of robot to an existing multi-agent system that is solving a particular task. Such a naïve robot could take a long time to become useful and may even disrupt the team dynamics, but if some of the older team members can get it up to speed it can be quickly made into a useful member. There are many such examples and although it will not be the focus of this thesis, research in this area can also have impact in other fields such as human-robot interaction. A robot and a human are

Figure 2 – Variety of robots used for search and rescue applications (Messina et al., 2005).

themselves widely different, and synchronizing definitions and establishing common ground has already been recognized to be important here (Stubbs et al., 2007).

An increasingly important application to robotics research is search and rescue. Within the United States, there is a significant effort to compile a list of standardized requirements and robot capabilities useful for the domain. Figure 2, taken from a preliminary report written by the program developed by NIST (National Institute of Standards) (Messina et al., 2005), shows a collage of some of the possible robots considered for emergency response. This figure underscores the fact that heterogeneity is inherently important for this domain, as one type of robot cannot possibly fit the requirements of a myriad of potential search and rescue scenarios. Furthermore, if different robots are to work together and communicate in such a scenario, understanding their underlying differences in action, perception, and representation will be crucial. For example, how would a small quick robot that can explore the environment efficiently but only has a crude camera describe victims perceptually to a more advanced robot with a higher-resolution camera?

1.2. Defining the Problem

As described, heterogeneity in robotics can stem from many different sources and at varying levels. This dissertation will demonstrate that one key component to reconcile such differences and perform successful knowledge transfer is to build an abstraction of low-level sensory data, and ground such abstractions in each robot through its own particular sensors. The notion of abstraction is already used in learning systems to improve generalization of learning and reduce the state space. One contribution of this dissertation is to demonstrate empirically that such sensory abstraction can indeed help both learning *and* knowledge transfer (Sections 3.10 and 5.6). In order to determine how to abstract sensory data, we take inspiration from Gärdenfors' conceptual spaces (Gärdenfors, 2000). Conceptual spaces is a cognitively-inspired multi-level representation that uses geometric spaces to represent concepts. This representation achieves good accuracy, can deal with uncertainty, and allows the problem of heterogeneity to be analyzed at multiple levels of representation. Using such a multi-level representation, we will define perceptual heterogeneity and pinpoint specific differences that can occur between robots at various levels and propose solutions for each.

Following terminology from conceptual spaces, we call the intermediate abstractions of low-level sensory data *properties*. We will define these properties formally, but for now they can be thought of as general categories of characteristics that objects have. For example, “blue” can be a color property and “large” can be a size property. In general, learning such abstractions in an unsupervised or unguided manner is a difficult open problem in robotics. However, they can be currently learned with the use of techniques such as scaffolding (e.g. Saunders et al., 2006) and providing supervisory signals. For example, the training will consist of learning lower-level properties (e.g. colors or object sizes) before learning concepts that combine them. One advantage of supervised learning is that we will be able to manually vary the training regime in order to vary the amount of overlap between two robots.

Once these representations are learned individually by each robot, we then develop methods that allow the two robots to learn models of their similarities and differences. We call this process *alignment*, and it requires the robots to interact in the world as well as to share *some* similarity, for example to share coordinate systems or properties that are used to describe objects. If a significant amount of these requirements are not met (e.g. in the case of a nanobot and a UAV), then the process of alignment will fail. Even in such cases, however, it is important to be able to decide whether there is sufficient overlap or to recognize that there is not. In our approach, we deal with uncertainty and cognizant failure throughout by measuring, for example, how much information has been lost in the transfer of knowledge between one robot and another given knowledge of which properties they share. Such considerations can then be used to determine whether the robots are too heterogeneous to be able to effectively communicate.

Note that several assumptions are implicit in this work. First, we assume that the robots operate in noisy, uncertain environments. This is usually the case when dealing with real-world robots. Second, as mentioned before, robots can vary not just in their perceptual capabilities, but also with respect to their motor capabilities, their goals, the tasks they perform, etc. In this dissertation we assume homogeneity with respect to these other characteristics when needed and are agnostic otherwise. For example, there is an implicit assumption in this dissertation that the robots' goals align, in that they care about learning about the same types of objects. Finally, we only study symmetric relationships between two robots. Other more complicated relationships, such as transitive chains, are possible. For example, a robot can transfer knowledge to a receiving robot, and the receiving robot can then transfer knowledge to a third robot. While studying such relationships would be interesting future work, they are not covered in this dissertation.

This leads to the research question at hand. Given a grounded perceptual representation, how can such knowledge be shared among perceptually heterogeneous robots? Since robots may differ at multiple levels (as discussed previously), we approach this problem in a hierarchical manner: robots, through joint interaction in the world, first determine what their differences are at the property level and then at the conceptual levels. One question this raises is the role of abstraction in communication and transfer. Will these property abstractions make learning and transfer easier? The process of alignment involves two-way interaction between the robots, either through joint exploration of the world or protocols where sensory data snapshots are exchanged. A key notion we introduce is that of a *shared context*, whereby robots view similar scenes and objects so that they can compare their resulting representations. Such constraints enable robots to reduce confounding factors in determining underlying robot differences such as differences in the parts of the environment that are being sensed.

After learning these models, the robots can then use them to exchange knowledge. Specifically, entire concepts can be transferred if the underlying representations utilize common properties. There are also several other ways in which the models of robot differences can be used. For example, the knowledge sharing itself must be sensitive to their differences, i.e. a robot should use difference models to decide what sensors, properties, and symbols to use when describing something to another robot. This sensitivity to capability differences is especially important during tasks requiring communication and coordination such as joint reconnaissance. Also, knowledge from robots that are more alike (according to criteria such as the amount of shared properties or concepts used for a task) should be considered with greater weight than robots that are substantially different. Hence, we explore the usage of learned similarity models in picking the most similar robot to communicate with.

1.3. Research Questions

The discussion above leads to the following research questions:

Primary Research Question

What interaction, dialogue, and adaptation processes are necessary to allow heterogeneous robots to model their differences, to use these models to exchange concepts and learning experiences, and how can such a system be used to improve the performance of a robot?

Subsidiary Questions

i. *How can robots model their differences in perception to improve their ability to communicate, and how can the establishment of a shared context help, if at all?*

ii. *What is the role of abstraction of sensory data in communication and knowledge transfer?*

iii. *What dialogues and protocols can allow two heterogeneous robots to use these models to align their knowledge and synchronize their symbols? How does the type of knowledge transfer possibly differ depending on the level of similarity that exists between the two robots?*

iv. *How can these models be used to make the knowledge-sharing and communication processes sensitive to the capability differences between the robots?*

v. *How can these models be used to pick peer robots that are more similar in terms of properties and concepts, for a particular domain of knowledge?*

1.4. Contributions

There are several contributions that have resulted from answering these research questions. They include:

- Demonstration that perceptual heterogeneity does indeed pose a problem for the transfer of learned object representations, even using modern computer vision algorithms that use object features specifically designed to be repeatable (Section 3.1).

  We have conducted an experiment using three robots with different cameras, twelve real-world objects, and a state of the art computer vision algorithm to explore the importance of learning with the robot's own embodiment and the effect of perceptual differences on knowledge transfer. We showed that even when using features, which are explicitly designed to be both repeatable and distinctive to particular objects, the highest accuracy is achieved when the robots use their own particular sensing to learn. Transfer from other robots can bootstrap learning, but can also result in catastrophic failures where the accuracy drops dramatically for certain objects due to missing features. This experiment demonstrated that, even in the best case scenario, perceptual heterogeneity can pose problems for knowledge transfer and that understanding the differences between the robots is important (Section 3.1).

- Demonstration that using abstractions of lower-level sensory data to learn can facilitate not only learning itself but also knowledge transfer, compared to using the raw sensory data to learn.

  We have conducted experiments using two robots with different sensors, thirty-four real-world objects, and a state of the art classification algorithm analyzing whether our method for sensory abstraction actually improves learning (Section 3.10) and/or knowledge transfer (Section 5.6). We demonstrate that it does improve both, especially when the underlying sensory data used by the robots differ (e.g. one robot uses an RGB color space while another uses an HSV color space).

- Algorithms and representations suitable for learning models of perceptual differences between two robots at multiple levels, utilizing sensory data obtained after the two robots achieve a shared context.

We use a grounded representation of properties (e.g. 'green') and their combination for concepts (physical objects, e.g. 'apple'), and demonstrate processes where the robots can explore an environment, locate objects, and build representations describing which properties or concepts both robots see simultaneously. In addition, we take potentially shared properties and further calculate how much information is lost when converting a concept from one robot's representation to another robot's representation, using information-theoretic measures. The resulting models represent which properties and concepts are and are not shared by the two robots. The representation is described in Chapter 3 and the learning of models of robot differences is described in Chapter 4.

- Protocols and algorithms for using these models for knowledge exchange in several scenarios: transfer of a concept unknown to one robot, choice of properties that will distinguish an object in the receiving robot's representation, and choice of one concept over another based on whether information will be lost by the receiving robot.

  We show how the models of similarities and differences between the robots can be used to perform these types of communication and knowledge transfer. We show how the models of similarities and differences between robots can be used to adapt existing knowledge by modifying the concepts to reflect missing properties. It is determined whether sufficient information is left to represent the concepts accurately. This adaptation occurs during the transfer of concepts, described in 5.3. For example, these are useful for search and rescue domains where one robot must describe an object's appearance to another. The transfer of a concept between two robots is described in chapters 5 & 6, while other types of communication are described in Chapter 7.

- Metrics for calculating similarity between two robots, based on their differences in capabilities and tasks.

We show how the models of similarities and differences between the robots can be used to estimate differences between two robots for a task. We take the set of concepts involved in the task and measure how much information can be potentially lost, and choose the robot partner that will result in the least amount of loss. This is described in Section 7.3.

- Implementation of the complete framework allowing heterogeneous robots to establish models of their underlying differences, use these models in a dialogue to synchronize their definitions and symbols, and exchange knowledge for a given task.

1.5. Dissertation Outline

Having described our research goals, we will describe background material and related work in chapter 2, and differentiate our contributions from the body of literature. Following that, we begin chapter 3 by demonstrating that heterogeneity does indeed pose a problem for the sharing of concepts, even in the best of conditions. With this motivation in hand, we then continue chapter 3 by detailing the representation of concepts used in this dissertation. The representation we use consists of multiple levels, allowing us to look at how two robots with perceptual heterogeneity can learn about their differences at these different levels. Chapter 4 begins this process by demonstrating how models of differences at the intermediate level of representation (properties) can be built using instances from each robot in a shared context.

Chapter 5 then describes a framework for transferring concepts between robots using the resulting models. The same chapter also demonstrates the value of abstracting sensory data in transfer and analyzes the estimation of information loss before transfer. In chapter 6, we introduce a more detailed information-theoretic metric that can be used to model robot differences. This includes a more detailed analysis of concept transfer for varying amounts of differences between the robots. Chapter 7 then uses the models of robot differences in order to perform additional tasks, such as describing objects in a manner sensitive to the other robot and picking the most similar robot to communicate with. Finally, chapter 8 concludes the dissertation and discusses the contributions made as well as future work. Relevant chapters contain simulation and real-robot experiments to validate our framework.

CHAPTER 2

BACKGROUND AND RELATED WORK

The topic of research in this dissertation touches upon many disparate fields, all the way from robotics involving learning by imitation and human-robot interaction to social-psychological theories of language formation and common ground. In this chapter, we review the literature in these areas and highlight key concepts that will be used to form the foundation of this thesis. A great deal of the work we review assumes homogeneity and involves agents in simulated worlds where sensor grounding and noise are not issues. In these cases, we note their assumptions and describe why they are not realistic for robotics problems. Where relevant, we also discuss differences from other research in robotics and what makes this dissertation topic unique.

The first concern in knowledge sharing among real robots is the use of symbols or sensor abstractions, and their grounding in the real world. Hence, in Section 2.1 we review the literature involving the symbol grounding problem, especially those that separate *physical* symbol grounding (Harnad, 1999) and *social* symbol grounding (Vogt and Divina, 2007). Social symbol grounding refers to the mutual grounding of symbols among multiple agents through local interaction and learning (Vogt and Divina, 2007). We focus on this aspect of grounding since we would like heterogeneous robots to share symbols and ground them to the same aspect of reality.

A large part of the problem in exchanging information is the correspondence of symbols in different agents, whether they are grounded or not. Although methods studying social symbol grounding may alleviate these differences if agents have collaborated for a long time, there are several problems with such existing work (e.g. (Jung and Zelinsky, 1999)). First, they do not address situations in which robots have already developed different vocabularies; this is a situation that is likely to exist as robots may, just as humans, interact with each other sporadically or on an as-needed basis.

Furthermore, they do not deal with heterogeneity, especially with respect to having different sensors or perceptual features. In a society of agents, common symbols are typically agreed upon in a self-organizing manner between agents through time. Hence, we also review research in fields studying the processes underlying ontological negotiation and language formation in Section 2.1.6. Some of these studies are in the field of multi-agent systems, but others study the psychological processes used during human-human interaction. There are some differences, however, from the problem posed here. For example, in heterogeneous systems not only can symbols themselves differ, but the agents cannot all see the same aspect of the world in the same way and hence the *meaning* of the symbols may also differ. In fact, some agents may not be able to see at all what others can, and hence they can instead negotiate to use overlapping sensors.

Even assuming that symbols are properly grounded and commonly shared, the problem of knowledge transfer is still not trivial. For example, there remain issues of when to ask for knowledge, how to distribute it, and how to resolve conflicts. In this regard, we review research into the transferring of knowledge among agents, mostly homogeneous, in the areas of multi-agent systems (Section 2.2.1) and distributed case-based reasoning (Section 2.2.2). We also discuss various methods of knowledge exchange used in robotics, especially implicitly via learning by imitation (Section 2.2.3).

In order to share knowledge in a particular conversation, the two agents conversing must share common ground (Clark & Brennan, 1991); that is, assumptions, beliefs, and most importantly context. This occurs via various dialogue mechanisms that ensure that both agents know what the other mean. When there is disagreement, it is expressed by the agent and worked out. In Section 2.3, we look at various psychological models of dialogue, especially those principles that can be used in robotics such as dynamic adaptation of detail level used depending on the other agent's responses. Although such negotiations are outside the scope of this dissertation, we will review the literature in this area due to its relevance.

In heterogeneous systems there is the additional problem that knowledge obtained from one robot has to be adapted to the receiving robot's capabilities, views, and existing knowledge. Although rare, we highlight the research that touches upon these problems. Finally, in Section 2.4, we explore literature seeking to characterize heterogeneity in a multi-agent system, mostly at a global team level.

2.1. Physical and Social Symbol Grounding

The symbol grounding problem refers to the problem of tying meaning to abstract symbols (Harnad, 1990). The meaning of "meaning" is itself controversial, but symbols should be tied in a way that is meaningful to the agent itself, be it objects or features in the world that it can sense or actions it can perform (affordances) (Gibson, 1977). In a multi-robot system, the problem becomes more complicated because not only do symbols have to be grounded, but they have to be commonly shared by convention in order to allow the sharing of information. This is called the social grounding problem (Vogt and Divina, 2007) and there has been some work analyzing how a group of simulated agents or robots can automatically agree on symbols via long-term joint interaction.

2.1.1. Physical Symbol Grounding

Coradeschi and Saffiotti have identified and formalized a problem similar to symbol grounding that they term *anchoring*: Here, the problem is "the connection between abstract and physical-level representations of objects in artificial autonomous systems embedded in a physical environment." (Coradeschi and Saffiottie, 2000). Vogt characterizes this problem as a technical aspect of symbol grounding, since it deals with grounding symbols to specific sensory images (Vogt and Divina, 2007). The symbol grounding problem in general deals with anchoring to abstractions as well, and includes philosophical issues related to meaning.

A popular class of representation for symbols, which is naturally grounded, is affordance-based representations. Here, the representation of objects (and the symbols they are tied to) consists of what the robot can *do* with the objects. For example, Deb Roy has studied the grounding of words by manipulator robots in both perceptual and affordance features (Roy, 2005b). He has also surveyed the area of language formation models that ground meaning in perception, and has noted the importance of future work in discourse and conversational models based on human studies for inspiration in aligning these models between communicating partners (Roy, 2005). In essence, this is the problem we are trying to tackle, and in addition to symbolic communication we utilize the ability of robots to jointly interact in the world.

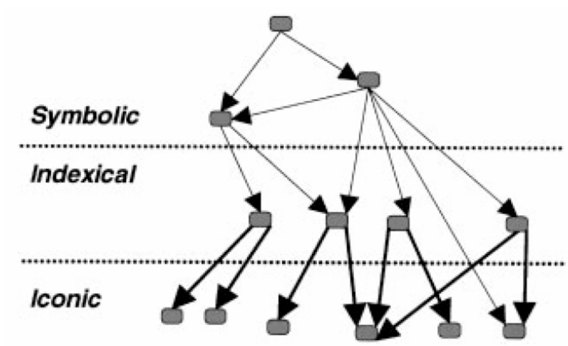


Figure 3 – Deacon's levels of representation (Jung and Zelinsky, 1999).

The aforementioned research looks into the grounding of symbols in a single agent or robot. We now look at multi-agent strategies for symbol grounding. The most similar research in spirit and implementation to this thesis is (Jung and Zelinsky, 2000). In this work, the authors implement a hierarchical system of knowledge representation among two heterogeneous robots and show that higher-level communication after establishing common symbols improves performance. The task they chose is a cleaning task, where one robot with a vacuum can gather large piles of garbage (in this case Styrofoam) and the other robot with a broom can sweep smaller pieces into large piles and reach into corners that the other cannot. They look at four levels of collaboration among the robots, requiring different levels of representation. They use Deacon's three levels of representations, portrayed in Figure 3 (Deacon, 1997). Iconic representations are those that bear physical similarity to the physical objects they represent, indexical representations are for correlations between iconic representations, and symbolic representations are for relationships between icons, indices, and other symbols. The four collaborative levels used are:

- No awareness of each other: The robot with the broom merely goes around and gathers small pieces into larger piles and the second robot can see these piles and vacuum them up.
- Implicit visual communication of likely litter positions: The vacuuming robot can see the robot with the broom, and hence likely positions of piles of dirt.
- Explicit communication of litter relative position: Here, the brooming robot sends information to the vacuuming robot about where the pile is located. This only occurs when the robot with the broom can see the other robot.
- Explicit symbolic communication of litter locations: Here, the two robots explore the environment together and develop a shared vocabulary of locations, and later while sweeping the robot with the broom can send directions to the other relative to these locations.

This work is interesting in many respects. First, it demonstrates the notion of different types of communications possible, ranging from none to explicit symbolic communication, and demonstrates the usefulness of the latter in the performance of a task. Our work directly relates to communication of grounded symbols. Second, it illustrates the levels of representations and the relation between symbols and internal representations nicely, with simple sensors and symbols. Finally, it is interesting in that heterogeneity of robots is overcome by exploring the environment together and having some similarity (namely, wheel encoders to represent locations relative to each other).

Even here, however, there is no notion of modeling or understanding their differences. Furthermore, long-term interaction is required to jointly develop each symbol before performance of the task. In our work, the robots can learn separately and will then interact to learn models of their differences and similarities. Also, in this case communication is performed regarding the actual task they are jointly performing, as opposed to more general knowledge pertaining to skills the robots may have. Despite being multi-layered, the representation is also very simple with odometry locations providing the shared grounding and symbols corresponding simply to their values: there is no implementation or transfer of a multi-level representation. Our work builds upon this significantly, among other contributions demonstrating how robots can figure out what features they share in order to provide shared grounding, and the learning and transfer of more complex multi-level knowledge.

2.1.2. Social Symbol Grounding

Symbols have to be agreed upon when there is more than one agent involved. In the example discussed above regarding symbolic synchronization among multiple robots, it comes about during the actual learning and grounding process that is performed jointly. Although this may be true of robots that experience the world jointly, this is not a realistic requirement in many situations (such as those in the motivation section) and hence we do not make this assumption. For example, it is likely that robots will be solving different tasks and gathering their own symbols and experiences, and these will have to be aligned after the fact. In this subsection, we will describe explicit methods dealing with the alignment of separate grounded representations learned individually. This is related to language formation and has been studied extensively in linguistics (Steels and Kaplan, 1999) and evolutionary or artificial life (Cangelosi, 2001),(Kirby, 2002). However, as we discuss below, some of the unique challenges posed by dealing

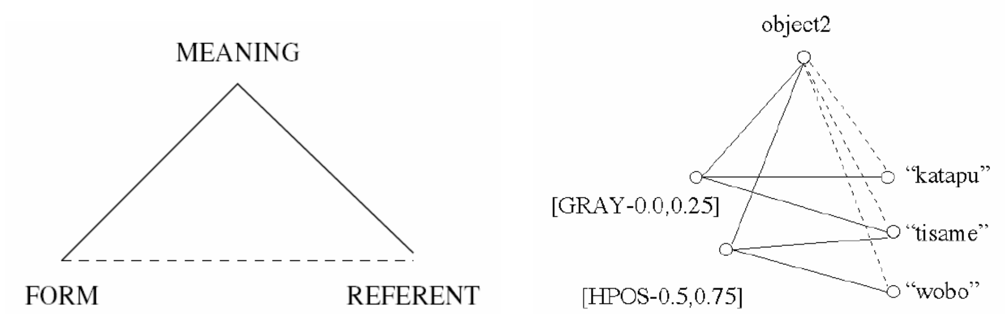


Figure 4 – Left: The semiotic triangle, referring to the relationship between meaning, form, and referent (Vogt, 2007). Right: A specific example of a semiotic landscape , showing co-occurrences between the three parts (Steels and Kaplan, 1999).

with the sharing of knowledge at *multiple levels of representation* among *heterogeneous* robots have not yet been explored.

Steels and Kaplan describe a "guessing game" in which two simulated agents view a common screen with various shapes and colors and attempt to obtain similar symbols to characterize them (Steels and Kaplan, 1999). They use what they call *semiotic symbols* (Vogt, 2003), depicted in the left portion of Figure 4. Here, instead of just having the symbol and meaning, there is a form (or word), the meaning (which can be a grounded aspect of the referent, e.g. color), and a referent (the object itself). The right half of Figure 4 depicts how there can be co-occurrences for multiple agents. For example, one agent may think "tisame" refers to the fact that the object in question has a certain color, while another agent may think the same word refers to its horizontal position. The problem then becomes to align these between the two agents.

The representation used is a discrimination tree, and is learned using *discrimination games*. The input consists of sensors that return continuous values, and feature detectors which return nominal values by splitting upon a sensor value (note that we use the term feature in a more general way in our work). Feature detectors can be nested hierarchically to essentially yield smaller discretized subspaces. This notion is slightly

Figure 5 – A depiction of the "Talking Heads" experiment (Steels and Kaplan, 1999) .

reminiscent of decision trees (Mitchell, 1997), except that they are limited to one attribute (the attribute value). Branches along the discrimination tree are mapped to symbols, providing a way to refer to specific discretization of sensors. For example, the symbol "upper-left" can be assigned to coordinate feature detectors, when the value is less than half of the screen for the width feature detector, and right of the screen for the height feature detector. The space of sensor values are dynamically discretized using the discrimination game. Essentially, multiple objects are given in a context, and the agent picks one of them. It then tries to find a feature that discriminates the target object from the rest of the context. If it is unsuccessful, it attempts to make it distinctive by creating a feature from a sensor that has none, or further refining an existing feature. If there are multiple features that could be used, heuristics such as picking the discretization corresponding to the discrimination tree with the shortest height are used.

Given these individual representations, agents then take part in a game in order to align their representations. The "Talking Heads" game, depicted in Figure 5, is meant to do this and begins by one agent issuing a symbol to the other given a context (in this case a board seen by both agents) with objects in particular locations with particular colors. The other agent must guess the specific object they are referring to. If the agent does not guess correctly, the answer is revealed by the initial agent and both of them attempt to update their discrimination tree representations of the symbol. They show that these types of games can allow the two agents to converge upon a similar vocabulary, and phenomena existing in human languages such as ambiguity or synonymy also occur.

While leading to insights into the evolution of language in a society of agents, there are several limitations to this work for robotics use. In many cases the language game is in simulation and even then required thousands of language games for alignment (Steels and Kaplan, 1999). Some implementations used cameras and simple color shapes (Steels, 2003), but the representations have thus far been used only for abstract shapes as opposed to real objects. Attempts to implement the system on real robots were met with mixed results (Nottale and Baillie, 2007). A more crucial difference, however, is that in all of the experiments the agents are homogeneous and have the same exact perceptual features. This dissertation focuses explicitly on heterogeneity. More importantly, the representation being aligned is one that discretizes sensors and assigns symbols to them; more complicated hierarchical trees are not used to describe the objects themselves (which are in fact not even represented).

Finally, in these earlier works explicit meaning transfer is not performed. However, if sensors or features are in common between robots, there is no reason not to transfer concepts explicitly. In our thesis, we build upon this to allow the robots to determine which features are shared by interacting in the world, and leverage this knowledge to perform explicit knowledge transfer for a multi-level representation that is richer. We also demonstrate additional capabilities such as adaptation of the representation to remove unshared features, capability-sensitive communication, and the ability to select robot peers that are more similar than others.

2.1.3. Symbol Grounding in Robotic Systems

As stated, there is little work on knowledge transfer among heterogeneous robots that have different learned representations *a priori*. Steels and Vogt have looked at implementing the previously mentioned language games on simple Lego robots equipped with light and infrared sensors (Steels and Vogt, 1997). Certain "objects" in the form of light events are observed, and then a teacher robot picks one of these, after which an alignment process (using infrared emitters and receivers) is performed so that they are both attending to the same object. The robots then pick discriminating features and assign symbols to them, and depending on whether that symbol is known to the robots or not an adaptation process occurs. There are also attempts to perform the Talking Heads experiment on an Aibo platform using vision (Baillie, 2004). However, the authors mainly discuss implementation issues of establishing shared context via shared attention on these platforms. The work seems preliminary and there are no results to date. The authors have noted that the simulated games take hundreds or thousands of trials, which is obviously difficult or impossible to do on real robots. Surprisingly, there is little work in the language formation process by roboticists that would be able to work out challenges in this line of research caused by dealing with real embodied robots.

Billard and Dautenhahn have looked at a situation involving two homogeneous robots where one teacher attempts to share its symbols with another via imitation (Billard and Dautenhahn, 1999). The learner follows the teacher while the teacher utters symbols based on what it is currently sensing. The symbols are based on sensing a light above the robots, and hence the learner robot senses more or less the same thing as it follows the teacher. It then makes associations between the sensor states and symbols using a recurrent network. During testing, the learner is then able to move to and ascertain the location of the teacher in a test scenario where the only information the learner has is the corresponding symbol. Even in this simple case, problems arising from differences between the learner and the teacher in terms of their location caused slightly mismatched vocabularies, however. Yanco and Stein (Yanco and Stein, 1992) earlier performed a similar study with small mobile robots, where vocabularies corresponded to actions instead of perception (such as turning left or going straight). The leader robot received reinforcement from the environment, while the follower robot receives a symbol from the leader robot. Reinforcement learning is used to perform the correct action, but the reinforcement for the two robots is linked in that they must both perform the correct action to receive positive reinforcement. Using this communication mechanism, the two robots are able to learn to perform synchronized action that is reinforced by the operator. In both of these examples of communication, the vocabularies were tied to simple, shared, features (or actions) between homogeneous agents. Again, there is no perceptual heterogeneity and no knowledge sharing at levels higher than features - key aspects of our thesis.

2.1.4. Shared or Joint Attention

Joint attention and the related notion of shared context has been cited as important in the social grounding of symbols in fields as diverse as psychology, language evolution, and robotics (Vogt and Divina, 2007),(Steels and Loetzsch, 2007),(Kaplan and Hafner, 2006),(Scassellati, 2002). The notion is that learners must be able to focus on relevant aspects of the world when a trainer communicates symbolically the name of objects. For example, the learners that followed behind the trainer in the preceding work aimed to establish a shared context so that similar perceptions can be grounded with the symbols communicated (Billard and Dautenhahn, 1999). In this dissertation, we utilize a notion of shared context not only for grounding during knowledge sharing, but during the learning of difference models between heterogeneous robots as well. Just as context plays a crucial role in disambiguating features that are relevant during symbolic communication, it is also useful in removing confounding factors or context in the determination of whether two sensors or features are similar or not.

We have performed some preliminary work in this area, including an analysis of different behaviors for ensuring a physically shared context in an object-recognition domain (Kira and Long, 2007), including an analysis of their trade-offs in terms of cost and the accuracy they afford in modeling differences between sensors on two robots.

2.1.5. Impact of Heterogeneity on Sharing of Multi-Level Representations

In the case where agents differ, there is an additional problem: agents will likely ascribe not only different symbols to different meanings in the world, but may also disagree on the meaning itself since they have different sensors with which to see the world, and the associated uncertainty. Robots may differ in their sensing and perceptual features, and must first determine which ones they share. We could not find any research focusing on this problem explicitly. Unlike abstract agents that have a fixed ontology available to them and no access to the data or processes that created it, robots have the opportunity of *action*. That is, two robots can jointly explore the world and together correspond the actual meanings of the symbols on which they are based. Of course, the extent and length of interactions are important, as current approaches in simulated or simple environments (such as those in (Steels and Kaplan, 1999)) appear difficult to scale to real robots since they do not use explicit knowledge sharing and require thousands of instances.

2.1.6. Ontology Alignment in Multi-Agent Systems

The study of multi-agent systems (MAS) has grown tremendously recently, both in the context of abstract or simulated agents as well as robotic systems. Within the field studying abstract software agents, there has been some work on the formation of consistent languages among agents and their negotiation of ontologies. Neches et al. advocated a large effort towards knowledge sharing among large knowledge bases, along with possible solutions such as having an intermediary language or separating domain-independent knowledge from domain-dependent (Neches et al., 1991). This problem is essentially an ungrounded form of what this dissertation is about; whereas such work attempts to align ungrounded symbols, we seek to align grounded symbols. Since that article, there has been a large body of work in this area, which is referred to as *ontology alignment* (Ehrig and Euzenat, 2004),(Laera et al., 2007). Here we will sample only a few.

An ontology defines a hierarchical representation of concepts and knowledge, in addition to other possible relations, and different agents (in this case abstract) may have differing ontologies. Unlike the research described in the previous subsection, in this case the symbols are not necessarily grounded and ontologies are aligned largely based on syntactic, semantic, and structural characteristics. An example of the problem can be seen in Figure 6, describing the structure of print media.

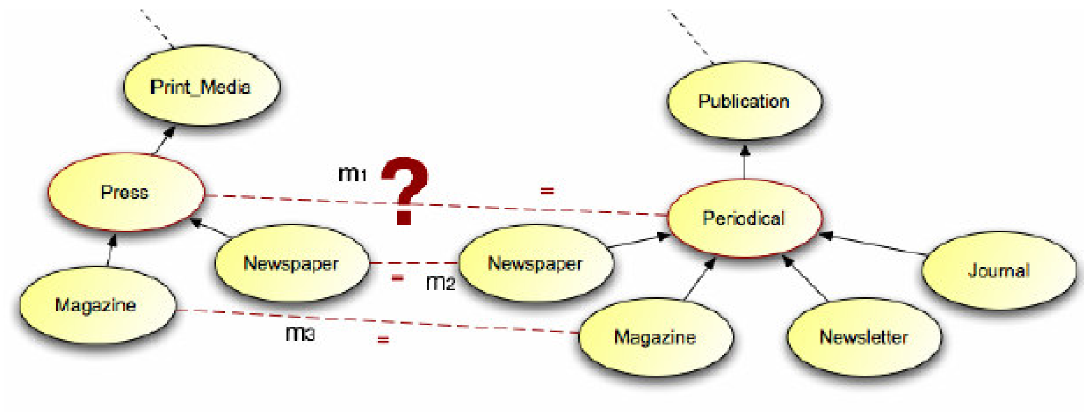


Figure 6 - Example of the ontology alignment problem.

The simplest method of alignment is to be able to convert each local ontology into some predefined global ontology by providing the mappings manually. Of course, this is inconvenient and there are many systems that attempt to perform automatic alignment, leveraging information such as the actual labels themselves (i.e. string matching), structural comparisons, or actual instances from ontologies obtained from users (Ehrig and Euzenat, 2004). For example, one can start out by defining all concepts with string similarity (based on similarity metrics such as hamming distance) as similar. One can extend this structure by defining two concepts in separate ontologies as similar if they have parents that are already deemed similar, or siblings that are deemed similar, or the paths from the concept to the root are deemed similar, etc.

Automatic alignment can depend on the purpose and context of an agent, and hence can also be done via argumentation whereby agents vie for their interests during the alignment process (Laera et al., 2007). Here, agents communicate the reason for deciding if a concept is similar (e.g. "because it has mapped parents"), and different agents can decide which set of reasons is acceptable. Alternatively, a fixed utility-based method can define a utility function based on all of the different methods possible (e.g. string or structural comparisons) and weight them.

All of this related research is conducted using abstract symbols and agents, and usually lacks concrete domains to which the schemes have been applied. There have also been decentralized approaches for agreeing on vocabularies via different strategies such as repeating symbols that are frequently used in a group (Diggelen et al., 2005). Here, different agents choose which vocabularies to use when communicating, with several strategies such as using the agent's private terms, the more recently received term, and the most frequently received term. Some of the methods take into account how many other acquaintances (i.e. that the agent has communicated with) the agent who used it have had. This work analyzes how different terms propagate depending on the strategies used.

The field of ontology alignment is interesting because at first glance it seems to bear upon the problem of knowledge sharing among heterogeneous robots. However, there are several important differences. In agent-based ontological alignment, the problem is one of aligning a hierarchy largely based on its structure and semantic similarity, rather than any grounded *meaning* of the symbols. In that way, the problem is unconstrained, where all that is available is a hierarchy of terms, possibly with instances from the hierarchy, and it is assumed that generally the agents have similar symbols for similar things but that they are grouped or named differently.

Unless robots acquire symbols jointly, this will not be the case in general. As such, this work differs from our problem where robots learn grounded representations, and hence joint exploration of the world (containing objects from which meaning is derived) is possible to negotiate similarities in features and symbols. Even without physical interaction, there is potentially an abundance of instances from the concepts that can be shared. There is greater possibility for verification of a mapping between symbols, for example via quizzing by directly showing the instances corresponding to that symbol. Furthermore, aligned ontologies often times describe meta-knowledge of particular *domains* (e.g. the print media in the example above), where there is inherent structure that can be used for alignment. In the case of *learned* grounded representations for a vast number of objects, such hierarchical structures may not be present or may be present for a large number of meanings leading to a great deal of ambiguity. Hence, mappings based on structure instead of meaning might not yield success. This is not to say that some structural mapping or other clever techniques from this field may not help, especially when more abstract concepts (such as the classification of animals into a taxonomy) are represented, but such high-level representations are beyond the scope of this thesis.

2.2. Knowledge Transfer and Analogy

Given a knowledge representation that each robot has built through experience, the problem of knowledge transfer comes into play. Here, there are problems of conflicting knowledge, differing knowledge levels, and adaptation of the knowledge to the receiving agent's capabilities. This dissertation mostly deals with the latter two issues only. We review work related to knowledge transfer in multi-agent systems and specifically in symbolic case-based reasoning systems. We then discuss work in knowledge transfer among robots, especially in the area of learning by imitation. Again, most of this work assumes relative homogeneity of the agents, and we review the few cases in which this is not an assumption. Finally, we remark on the limitations of these previous research efforts and specific problems arising from heterogeneity that we will address.

2.2.1. In Multi-Agent Systems

The study of collaboration among multi-agent systems can be categorized into several methods (Weiss and Dillenbourg, 1999). *Multiplicative* mechanisms involve separate learning among independent agents, where their interaction may affect sensing but does not affect the learning processes themselves. *Division* mechanisms refers to the division of tasks among agents that may or may not have differing capabilities. Finally, *interaction* mechanisms are those that involve active interaction during learning that affects the learning processes. This may involve not just raw data exchange, but knowledge exchange at multiple levels. It also requires an added layer of complexity in terms of processes for argumentation, conflict resolution, and establishment of a common grounding. This latter form of interaction is the one we are interested in.

One subfield of research fitting this characterization is *negotiated search*, where multiple agents cooperate in searching for solutions and argue over conflicts (Lander and Lesser, 1992). Many times, different agents work on different aspects of the problem, and they collaborate on achieving global consistency from their local solutions. Lander and Lesser looked at two negotiation strategies: *extended search* whereby the agents extend their local searches to include new solutions if there is a conflict, and *relaxation* whereby an agent relaxes some of its solution requirements. The authors also mention the problems of dealing with heterogeneous agents that have differing representations, but in the end decide not to resolve any conflicts in knowledge. Furthermore, the problems being tackled here are different than the ones we focus on in this thesis. In negotiated search, there is a multi-agent system working towards the same goal, and individual agents must negotiate their constraints. In our problem, we are dealing more with sharing knowledge that may be relevant to each individual robot's own goals.

In many ways, negotiated search has a flavor of the task allocation problem, where there is a heterogeneous team with many different capabilities and the problem is to give the proper problems to the best agent. This is a well-studied problem in robotics (Gerkey and Mataric, 2004),(Ulam et al., 2007), and will be discussed towards the end of this chapter. While usually there is some modeling of agent capabilities in this field, the focus is not on understanding the differences between individual agents and how they affect each agent's knowledge (since once a task is assigned, there is no communication of knowledge for solving the actual task *per se*).

Other work in multi-agent systems that is relevant includes work on negotiation of actual knowledge via logical arguments (Kraus et al., 1993). Again, agents are assumed to have homogeneous representations with no uncertainty as to what the symbols mean. Also, there is a great deal of work in determining confidence or trust when interacting with other agents (e.g. Becker and Corkill, 2007). These are based largely on probabilistic representations of previous experience, and they look at how to use confidence measures when integrating contributions from different agents. In this thesis, we will look at how the characterization of the differences between multiple robots sharing information can influence the choice of which robot to collaborate with.

2.2.2. Knowledge Transfer In Case-Based Reasoning

There is also research into distributing knowledge specifically in an instance-based learning framework, in the context of case-based reasoning (CBR) (Britanik and Marefat, 1993),(Prasad et al., 1995),(Nagendra et al., 1995),(Watson and Gardingen, 1999),(Ontañón and Plaza, 2001),(Leake and Sooriamurthi, 2002). Instance-based learning is a lazy learning method in that it delays most of the processing until query time. It is memory-based, where key instances are stored that were either directly experienced or adapted due to experience. To solve a new problem, a subset of these

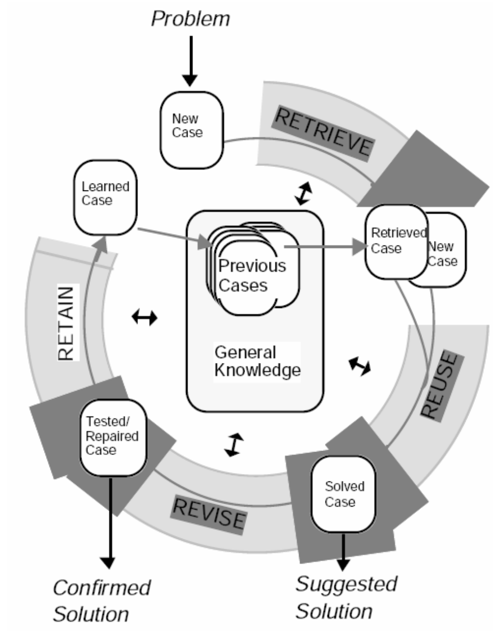


Figure 7 – The Case-Based Reasoning cycle (Aamodt and Plaza, 1994).

instances is used to approximate the target function locally, and their solutions are somehow combined to solve the new problem. For example, one method could be to use k-nearest neighbors, where the solution of the *k* closest points (as defined by a similarity metric) are averaged in a weighted manner based on their distance to the query point (Mitchell, 1997).

Case-based reasoning is a specific form of instance-based learning with a processing cycle depicted in Figure 7 (Aamodt and Plaza, 1994). Upon receiving a new problem, the case retrieval processes obtains a set of cases and picks the best among them. The solution to this case is then reused, but is revised or adapted to the new problem. It is then evaluated upon its use, and retained with information about how good the solution was.

Among the first researchers to study distributed case-based planning are Britanik and Marefat (Britanik and Marefat, 1993). They discussed a CBR system that, upon failing to find a solution locally, would query other systems. If a remote system finds a case that is more similar to the problem than the local system, it sends its solution back. To deal with heterogeneous representations, an interpreter module is introduced that maps local symbols to a global vocabulary. This entire process and mappings are defined by hand. This dissertation seeks to deal with the problem of mapping grounded symbols by taking advantage of the fact that the two robots can explore the same environment.

Ontañón and Plaza, among other colleagues, have also explored the problem of a multi-agent case-based reasoning system (Ontañón and Plaza, 2001),(Ontañón and Plaza, 2002),(Ontañón, 2005). Their early work looked at two modes of cooperation among case-based reasoning agents: **DistCBR** where problems are transmitted and the receiving agent solves the problem and sends back the solution, and **ColCBR** where in addition to the problem, the solution method is also transmitted (which is then run locally on the receiving agent's knowledge base). The problems were sent to either all other known agents, or to each agent one by one until an acceptable solution was found. The focus in their work was mainly on adding distribution semantics to a representation language called ***noop***. These extensions allowed for *mobile methods* consisting of entire task decompositions that can be sent to the receiving agent.

In later work (Ontañón and Plaza, 2001), they related this problem to ensemble learning in the machine learning community, for example, bagging predictors or boosting (Bauer and Kohavi, 1999), although there are some important differences such as lack of central access to all of the data. Issues relevant for use in an industrial setting were also looked at, including issues of security, scalability, and privacy of the cases. Instead of swapping actual cases, each system allows a CBR agent to use the cases internally and vote on a solution, but does not make the actual cases used to solve the problem public. Hence, this work distributes the *reuse* process, where use of the cases to solve problems is distributed. There is also work by the same research group in distributing the *retention* portion of the cycle, by deciding when to offer a case as well as when to retain a case locally (Ontañón and Plaza, 2003). Figure 8 depicts this process, with two strategies for

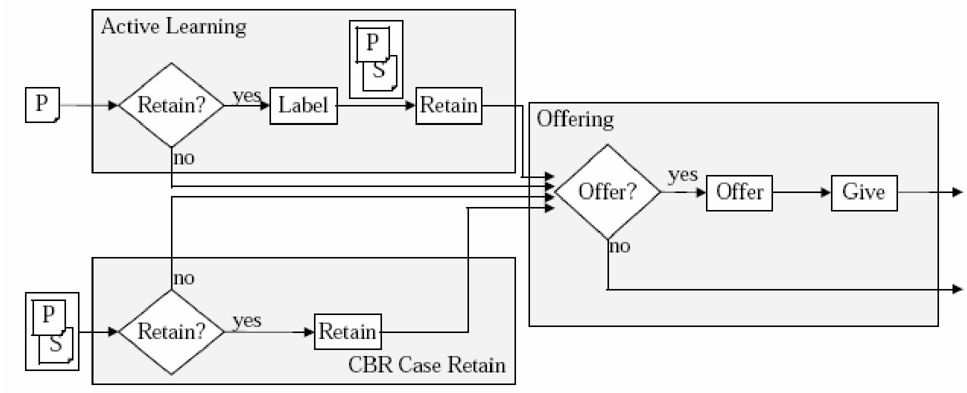


Figure 8 – Depicts a distributed retention policy, with two strategies for deciding whether to retain a case (left) an one deciding whether to offer the case to another agent (Ontañón and Plaza, 2003).

deciding whether to retain a case received by another agent (based on either ability to solve the problem, or disagreement of the solution if the problem *can* be solved) and one strategy for deciding whether to offer a case (based on whether it has been retained locally or not). Later, we will see other work that distributes the *retrieve* process.

Given a problem, the local CBR agent communicates with its peers and obtains a solution endorsement in the form of $\{(S_k, E_k^j)\}$, where S_k is one solution from a finite list and E_k^j is the number of cases endorsing the solution. Several collaboration schemes are used, such as asking all agents and summing the votes or only asking agents until the confidence in the solution reaches a certain threshold. The confidence metric used in this case is calculated by taking the discrepancy between the majority and the rest of the votes, and was also learned via decision trees. They also looked at bartering cases, both in an unrestricted manner and by a token passing mechanism (Ontañón and Plaza, 2002). Results indicated increases in performance due to bartering, but there was no research evident into how to select which cases to barter and whether it is better to get as many cases as possible or only the key cases that are missing.

More recent work by this research group has involved argumentation and counter-examples in the knowledge exchange process (Ontañón and Plaza, 2006). Specifically, they use a CBR system where cases are tagged with *justification*, or predicates that

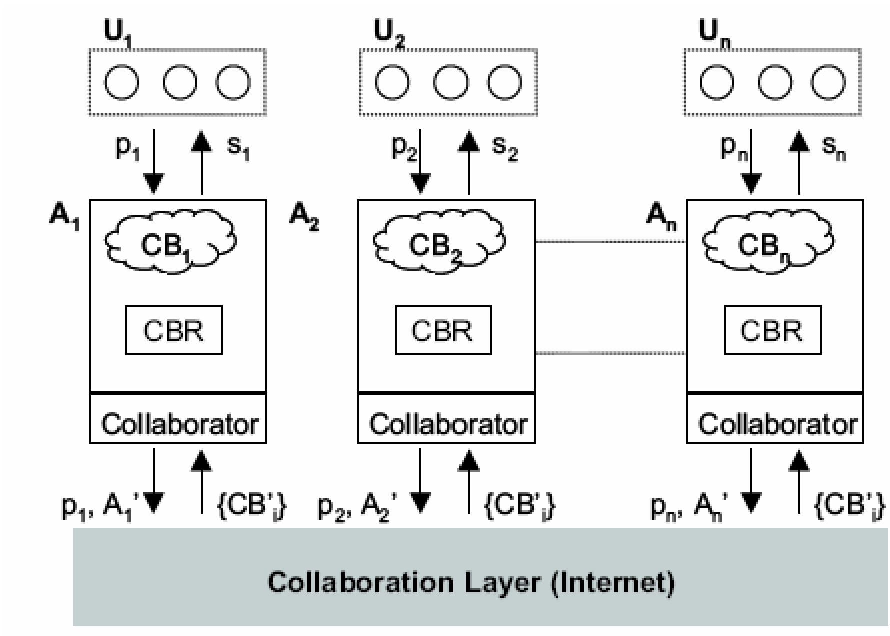


Figure 9 - Collaborative Case-Based Reasoning architecture (McGinty and Smyth, 2001).

explain why the case votes for a particular solution. In this case, the learning method has to support the creation of these justifications from experience. Again, confidence measures are used by taking into consideration the number of cases that agree with the justification (i.e., agree on a solution when the justification clauses are true). If two agents disagree on a solution, they enter an argumentation phase where the agent proposing the solution with a lesser confidence has a chance to rebut. Rebuttal takes the form of a counterexample, which is a case with a justification that is *more specific* than the one being argued about. This criteria is taken from literature studying argumentative logical deductive systems. The process can iterate with multiple example/counterexample pairs, and if any agent sends the same argument twice the process halts in order to avoid infinite iteration.

Again, the two agents have disjoint cases; that is, no two agents have the same case (cases are obtained directly from the training data). In our work, we use a dialogue in which the robots can either exchange instances directly, or jointly explore the world (an ability unique in robotic systems as opposed to abstract agents). We do not make use of argumentation as it is outside the scope of the dissertation, but such a process may be useful and is left as future work. (McGinty and Smyth, 2001) have explored a collaborative case-based reasoning system in the domain of personalized route selection. The basic architecture is shown in Figure 9. Cases are routes that are known to be preferred by the user from past experience (i.e., the user has taken the route before). When a route in an unexplored region of the map is encountered, the local CBR system cannot give a solution because it has no existing data for that region. Hence, the case library from other users is queried, with a quality measure attached to the resulting case. An interesting twist in this work is that the confidence is based on the coverage of the case library in addition to the similarity of the two users. The similarity metric is calculated by finding similar problems in their case libraries, and then evaluating the similarity of their solutions. Since in this case the instance-based learner is attempting to approximate a user model, data from users that have similar tastes are more pertinent. This is obtained by finding similar regions of the map for which the two libraries share data, and seeing whether their solutions agree. Other examples of similar research in distributing the querying process include (Leake and Sooriamurthi, 2002),(Prasad, 2000),(Hayes et al., 2005). The notion of using similarity in the process of retrieving knowledge from peers is one we will share in our work, although our metric will be based on the similarity of the robots.

2.2.3. Knowledge Sharing In Robotic Systems

Related work in robotics with respect to communication looks at, for example, maintaining belief in a multi-robot team (Khoo, 2003). The approach used here is to aggregate behaviors where team members share their data, allowing them to have the

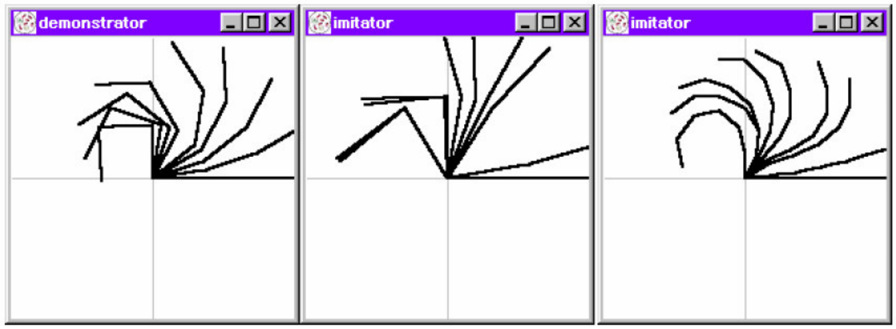


Figure 10 – Manipulator imitation across different embodiments (Alissandrakis et al., 2002).

same picture of the environment. To our knowledge, there is no work to date involving explicit knowledge transfer among heterogeneous robots, while understanding how such heterogeneity might affect these processes. Often communication involves symbols defined by the designer and not grounded in perception, or even if it is, then it is grounded in features that are assumed or known to be shared. This is also true for multi-robot solutions to the problem of SLAM, where the designers *a priori* decide on common representations (Thrun and Liu, 2005). Our work attempts to extend this area by tackling the problem of knowledge exchange and symbolic communication, but where the knowledge is *grounded* and hierarchical. Furthermore, we deal with the challenges of having heterogeneous robots that may only share a subset of sensors or perceptual features.

Learning by imitation is another heavily studied subfield in robotics that involves knowledge transfer but where there is no direct communication of explicit knowledge. It is a form of learning that involves the transfer of knowledge among robots by actually performing the target task. There is some research into learning by imitation across different embodiments (Alissandrakis et al., 2002),(Dautenhahn and Nehaniv, 2002),(Alissandrakis et al, 2003). The major approaches use task-based metrics of success, optimizing these on new embodiments during imitation. This side-steps the issue of heterogeneity in that it is not modeled or reasoned about. For example, the difference between optimization of end-points, trajectories, or entire paths for a given task is looked at. Domains used include chess, where the pieces differ in their movement capabilities, and a robotic manipulator (in simulation) with different degrees of freedom. The latter example is shown in Figure 10, where manipulators having lower degrees of freedom (two left examples) attempt to imitate a more articulated manipulator (right example).

In this case of learning by imitation, knowledge transfer is limited to physical movement or other actions that can be seen by the other robot, and knowledge is task-specific. There are other differences between this research and our dissertation research. First, there is no attempt at understanding the differences of the two robots; the knowledge transfer is abstracted by tying it to the task metrics itself. Learning by imitation encounters the problem of heterogeneity in points of view and in some cases embodiment. In this thesis, we study these aspects of heterogeneity more deeply in addition to other forms of heterogeneity stemming from having different sensory *groundings* for knowledge. We deal with perceptual heterogeneity, while in learning by imitation the more significant problem is motor heterogeneity.

2.2.4. Unexplored Territory

The sharing of knowledge among *heterogeneous* robots acting in the real world presents unique challenges and problem characteristics that have not been addressed by this body of literature. A major assumption of the large majority of this research is homogeneity in terms of the agents, learning methods, symbols, and representations. In this work, we begin to get at the heart of the problem caused by *differences* between the robots, in terms of sensing and perceptual capabilities, symbols and definitions, and levels of knowledge. For example, if robots were to exchange cases in CBR, it would not be as simple as sending the cases from one robot to the other, since disagreements in solutions might arise due to these differences, and adaptation to the receiving agent must be performed. Also, the usual issues of noise, limited frequency of interaction, and bandwidth that come up when using robots are also largely ignored in the agent-based work. Obviously, some heterogeneity results from using real-world robots as well, for example through differing points of view.

Furthermore, robotics provides an additional aspect of *action* available for the solution of these problems. While some research has looked into using dialogue to synchronize symbolic representations, robots can use *joint exploration* of the environment as well. In other words, robots can actually target specific regions of the problem space in order to home in on their strengths by virtue of their differences, although the cost of so doing must be traded off with the resulting gain. Finally, assumptions such as privacy and inability or undesirability of swapping entire cases are not pertinent. Although it is beyond the scope of this dissertation, our framework would allow much of the preceding work to be implemented on real robots. Instead of assuming common symbols, our methods can be used to learn the mappings between symbols grounded differently by different robots.

2.3. Context, Common Ground, and Application to HRI

In order for knowledge to be shared in a specific conversation, there must be some common ground between the two agents. Common ground consists of mutual knowledge, beliefs, and assumptions (Clark and Brennan, 1991) providing a foundation for knowledge exchange and communication. Most of the time common ground is engineered in robots to exist by assuming a shared task, representations, and experiences.

Even then, to share solutions for a particular context, that context must be understood by both agents. This dissertation deals with the establishment of a common ground with respect to the sensors and features used by the robots, by allowing robots to explicitly model their differences. While only scratching the surface, this capability can potentially provide the foundation on which to explore a fuller notion of common ground. We now review psychological studies of the establishment of common ground among humans, mainly through dialogue. We also discuss applications to human-robot interaction, where large differences exist between the two interaction agents (the human and robot), and how the establishment of common ground has been shown to be important. We conclude by remarking on differences between human-robot interaction and robot-robot interaction during collaboration.

2.3.1. Psychological Studies of Human Dialogue and Common Ground

There has been a great deal of study into how humans use communication to establish a common ground during collaboration (Clark and Brennan, 1991), as well as studies of how to use various forms of communication media to help when the collaborators are not co-located (Kraut et al., 2003). Herbert Clark and colleagues, in particular, have looked at properties of human dialogue when people attempt to make sure that they understand each other. They state that the process requires the establishment of mutual knowledge, mutual beliefs, and mutual assumptions (Clark and Brennan, 1991a). Another principle is that of least collaborative effort (Clark and Wilkes-Gibbs, 1986), which posits that people attempt to minimize the effort of their dialogues (i.e., how many words or phrases are used) based on the dynamics of interaction. If something is agreed upon previously, fewer phrases are used because it is understood to be known by both people. There are also notions of coming up with accepted references or points of view, and what happens when one person disagrees with the other's point of reference and has to correct it.

For example, they looked at a task that made two collaborating participants agree on descriptions for abstract figures. As the trials proceed, the two participants use less words and agree upon terminology for describing the figures. There is also work on how experts collaborate with novices, and how each person assesses the other's level of knowledge and adjusts their descriptions accordingly (Isaacs and Clark, 1987).

Although we have not pursued this in this dissertation, these psychological studies of human dialogue can provide insight into knowledge sharing among robots. For example, using the principles of *least collaborative effort* will allow robots to exchange information in a manner that is sensitive to the amount of assumptions and context they have in common; if they both have similar assumptions, then a smaller number of symbols or features may be communicated. Our framework can potentially help here by modeling what similar properties are shared between robots. Although bandwidth in robot communication is much larger, with large knowledge bases this principle will be important. More importantly, if one robot does not agree with or share the context communicated, then a dialogue process must ensue where it can correct the other robot or ask for more information. Robot communication is often engineered to be fixed in terms of what is communicated, but this can be brittle and inefficient. There has been little work on dialogue processes ensuring that the two robots share common ground before sharing knowledge about a particular context.

2.3.2. Applications to HRI

Human-robot interactions is a recent field that aims to study how humans interact with robots and ways of making this interaction more effective (Fong et al., 2003). It has been acknowledged that common ground and similar concepts are crucial in such interactions

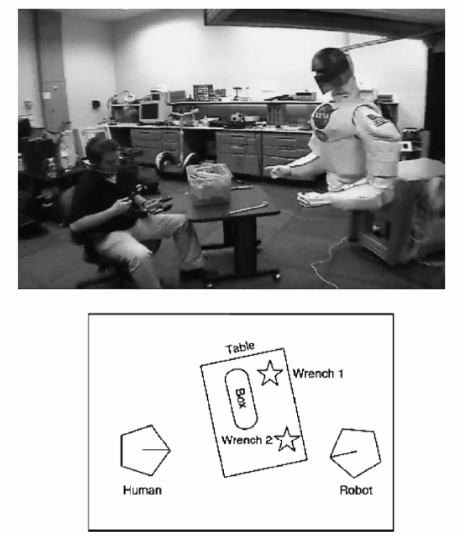


Figure 11 – Perspective-taking task using the NASA Robonaut (Trafton et al., 2005).

(Stubbs et al., 2007),(Kiesler, 2005). Stubbs et al. performed a field study of scientists and engineers interacting with a Mars-like rover that explored the Atacama desert remotely (Stubbs et al., 2007). Two different studies were performed, one where the robot was mainly teleoperated and another where the robot had some autonomy. They found that the lack of common ground, especially since the robot was remotely located, caused many of the miscommunications.

In order to work together, a human and robot must share contextual information in addition to knowledge regarding the robot's decision-making. Which one is more important depends on the level of autonomy the robot has. These field studies have shown that for weakly autonomous systems, where planning and acting is mainly performed by human users except for low-level functions such as obstacle detection, common ground in terms of perspectives and sensing is crucial. When the robots are more autonomous, it becomes important to understand the robot's decision making as well. The researchers recommended the addition of perspective taking skills. Some work in this area exists already, e.g. (Trafton et al., 2005), depicted in Figure 11, which looks at perspective-taking in a task where the robot must grab a wrench for a person across the table. The researchers also recommended allowing the robot to expand its vocabulary in terms of objects, locations, etc. so that there is more common ground. They implemented the perspective model as production rules in ACT-R, a cognitive architecture that uses a production system to model human cognition (Anderson, 1993). Furthermore, robot feedback in the form of dialogue was also lacking. Hence, they recommended seeking inspiration from studies of common ground, such as those cited in the previous subsection, to allow the robot to expand the amount of common ground they have with humans.

2.3.3. Differences Between Human-Robot and Robot-Robot Interaction

Although similar, there are several important differences between knowledge exchange in human-robot and robot-robot situations. First, current HRI research deals with interaction between humans and robots that do not have much in common *a priori*. For example, humans are endowed with an enormous amount of knowledge from socialization and learning. Most of the language vocabulary is already shared. Humans also have certain preconceptions of robots, which have proven to be important when considering their interaction with robots. The robots that humans interact with, on the other hand, do not have a great deal of general knowledge about the world, especially from the perspective of humans. This means that there is a huge imbalance between the human and the robot. Also, in most of the human-robot interaction studies, it is assumed that the robot is serving the human and hence should adapt to the human's capabilities. In multi-robot scenarios, the robots are more similar in status and which robot's preferences should be overridden is context-dependent. Naturally, as human perception of robots changes and as robot learning becomes more adept, the interaction dynamics will change. When robots interact with *each other*, on the other hand, they may have a similar level of ignorance in terms of world knowledge, at least relative to a human-robot disparity. They also may share the same learning and developmental trajectories, with periodic communication between them (although this is not necessarily the case).

A second difference is that the medium with which a human communicates with a robot differs significantly with the medium used by robots to communicate with each other. The medium defines what and how much can be reasonably communicated (Clark and Brennan, 1991). For example, humans can use language and gestures to communicate their internal states in a heavily distilled and abstracted form. The efficiency and bandwidth of this communication is rather low. Robots, on the other hand, can easily communicate even the raw sensor values they are seeing, and communication bandwidth may be extremely large. Furthermore, robots can share entire experiences in the form of key sensor events with each other, in effect allowing the receiving robot to actually "experience" the world for itself, without having been there at the time. In this dissertation, we actively leverage the fact that real-valued data (for example, representing concepts) can be easily transferred between robots. We use this both to learn models of robot differences as well as for direct knowledge transfer.

At the same time, our framework utilizes intermediate representations (properties) that are more general and compact than raw sensory data. This substantially reduces the bandwidth required between the robots to communicate. Hence, the amount of bandwidth needed is somewhere in between the transfer of raw sensory data and purely symbolic communication as performed by humans.

Research into learning methods via a human teacher and HRI can be combined with our work, allowing a human to teach a robot which can then share this knowledge with other robots. Such a mechanism can obviate the need for the human to expend additional time and effort beyond training the first robot, despite differences among the robots themselves. Hence, our work is applicable to many situations in which robots have to teach each other, whether there are humans in the loop of teaching at some point or not. We utilize the advantages that two communicating robots have (such as sharing of sensory data), and thus it is well suited to the R-R (robot-robot) parts of the interaction.

2.4. Defining or Characterizing Capabilities and Heterogeneity

As stated previously, there is a dearth of research into understanding how sensory, motor, and knowledge heterogeneity at the individual robot level can be modeled to improve communication between heterogeneous robots or how these differences affect the problems of knowledge transfer. There is limited work in creating taxonomies of robot capabilities (Messina et al., 2005),(Matson and DeLoach, 2003). NIST is interested in this for advancing capabilities for search and rescue domains, and the analysis is done by the organization with respect to what capabilities are suitable for this specific domain (Messina et al., 2005). Matson and DeLoach use a capability taxonomy to distribute tasks and organize teams, and their taxonomies are hand-modeled and at the sensor level. To our knowledge, no one has worked on characterizing sensor and feature level differences, for the purpose of knowledge sharing between heterogeneous robots.

There exists research relating to heterogeneity at a *global* level in terms of the task to be performed. For example, Balch explored the use of social entropy as a measure of heterogeneity, mainly at the behavioral and capability level (Balch, 1998). There is also research into how heterogeneity at this level affects, for example, the task allocation problem (Ulam and Balch, 2003) or learning in teams (Balch, 2005). Parker has dealt with heterogeneity both from the perspective of a small team of robots measuring each other's performance capabilities (Parker, 2000), as well as large teams in which tasks must be performed cooperatively (Parker, 2003). In the work involving small teams, changes in the performance abilities of teammates was constantly monitored, and inability to achieve goals spurred the other robots to take over the tasks. This deals with heterogeneity at a local level, but abstracts it in terms of task performance, not the actual modeling of difference. This is similar to the work on learning by imitation mentioned earlier (e.g., Alissandrakis et al., 2002). In the case of large heterogeneous teams with distributed capabilities, heterogeneity affects how the robots must work together, but there is no explicit symbolic communication between them and hence problems caused by heterogeneity with respect to knowledge sharing and grounding are not addressed.

The modeling of heterogeneity at the local level of individual robots is a natural first step to explore in order to allow robots to share knowledge. In the next chapter, this serves as our starting point, after which we look into how to use these models to transfer concepts between robots, communicate in a capability-sensitive manner, and determine the amount of differences between robots.

2.5. Summary

In this chapter, we reviewed a selection of related work studying social symbol grounding, language formation and ontological negotiation, knowledge transfer, common ground, and heterogeneity. The majority of the work has underlying assumptions of multi-agent homogeneity or in some cases lack grounding in the noisy sensory data that must be dealt with in robotics; we posit their assumptions to be unrealistic for general robotics use and have described problems raised by heterogeneity at each level that we seek to address. Furthermore, in many cases it is assumed that mappings between symbols in different agents are known *a priori*, for example during the sharing of cases in a case-based reasoning system. Since our framework deals with learning these models on real embodied robots, it can potentially facilitate the application of the related work to real robots.

A common theme in the synchronization of vocabularies, transfer of knowledge, and establishment of common ground is *interaction*. In agent systems, this involves argumentation, negotiation, or justification via direct communication. Although these mechanisms will undoubtedly be useful in robotic systems as well, there is potential for more, namely joint exploration of the world. Counter-examples can take the form of actually leading the other robot to a situation that contradicts its knowledge. Although useful, this form of interaction is much more costly, and hence the increase in certainty must be traded off with this cost.

As a summary, unique characteristics of our dissertation research include:

- Explicit learned models of robot differences *at multiple levels*, built via joint interaction in the world.
- Methods for abstracting raw sensory data to buffer underlying sensor differences between robots. We show evidence that this abstraction does indeed help during learning and knowledge transfer.
- Use of low-level difference models to determine when to exchange structured knowledge; our algorithms modulate the amount and type of communication based on the level at which the robots differ.
- Estimation of information loss when transferring concepts from one robot to another.
- Use of similarity models to adapt knowledge so that it is understandable by the receiving robot and calculate metrics used to measure how different two robots are.

CHAPTER 3

CONCEPTUAL SPACES: A GROUNDED CONCEPT REPRESENTATION

We begin this chapter with an important experiment that presents evidence for our motivation that perceptual heterogeneity does indeed present a problem for knowledge sharing. This experiment explores the importance of learning with a robot's own embodiment as well as the importance of modeling differences between robots for the purpose of knowledge sharing. We do this using a best case scenario, where a modern computer vision algorithm and perceptual features explicitly designed to be repeatable across images are used. With this motivation in hand, we then introduce our multi-level grounded concept representation inspired by conceptual spaces (Gärdenfors, 2000). One advantage of using such a multi-level representation is that it allows us to define and explore perceptual heterogeneity in robots at *multiple levels of representation*. We present this definition, after which we describe our experimental platforms and scenarios. Finally, we perform evaluations demonstrating that this representation can be successfully learned with experience and that our use of intermediate abstractions aid both learning and the transfer of knowledge.

3.1. The Problem of Heterogeneity: A Motivating Experiment

In chapter 1, we presented the underlying motivation for the research conducted in this dissertation. Robots can differ perceptually, and these differences can present problems when robots need to transfer knowledge or effectively communicate. A natural question

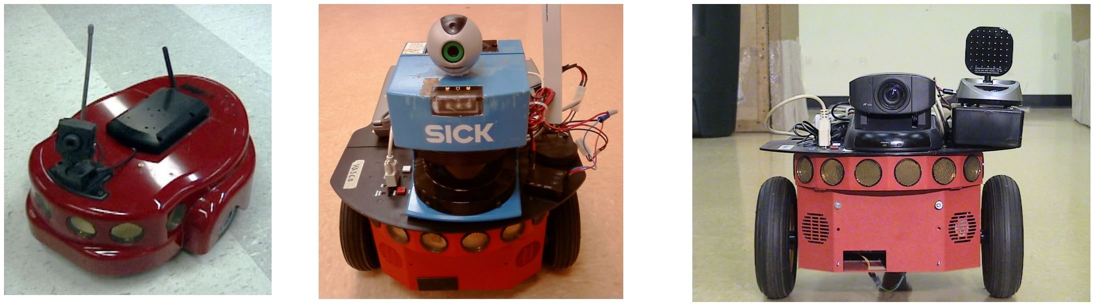


Figure 12 – The three robots used to demonstrate the problem of perceptual heterogeneity.

for a skeptic to ask based on this motivation is: How important is it that a robot learn with its own sensing? Can't a robot simply transfer representations it learns to another robot, no matter how different? In other words, how much of a problem is perceptual heterogeneity *really*?

Certainly, we will answer such questions using our representation throughout this dissertation. A question that remains, however, is whether other representations can be immune to such differences. Specifically, one approach in computer vision is to use feature detectors that are repeatable across variations in lighting, scale, or other transformations. These feature detectors find unique patterns in images that can be found frame to frame, and image patches around these features are then used to represent these features. The image patch descriptors are designed to be unique, in the sense that patches around features on other objects or scenes should not be similar. While this representation is limited only to vision, we believe that it represents a "best case scenario" for testing whether knowledge transfer can occur without modeling robot differences. In other words, we believe this representation is not as likely to fail across multiple robots, given that these features are designed to be both repeatable and unique.

Figure 14 – Respective images from the three robots above. Left: Amigobot. Middle: Pioneer 2DX with a web camera. Right: Pioneer 2DX with a wireless camera.

Figure 13 – Twelve objects used in the experiment.

The experiment we conducted used three different robots: a Mobile Robots Amigobot and two Pioneer 2DX robots. The robots are shown in Figure 12, and their respective images of similar scenes are shown in Figure 14. As can be seen, the cameras differed in their color properties, blurriness, and position on the robot (resulting in perspective differences). Each robot obtained approximately one hundred images of twelve different real-world objects, shown in Figure 13. Fifty of these images were randomly chosen and

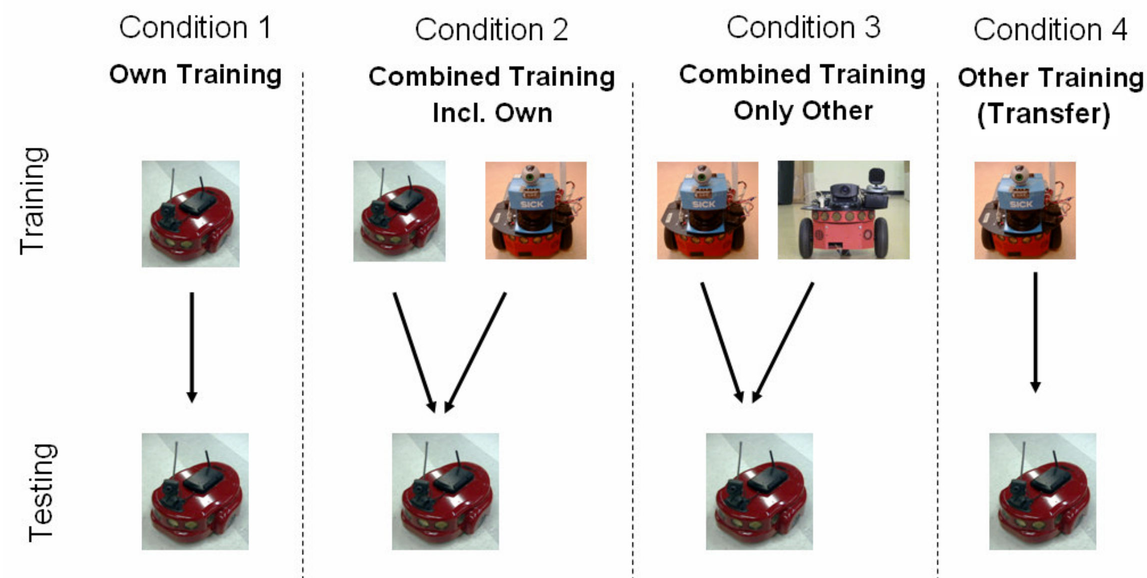


Figure 15 - Four conditions used in experiment. Condition 1: The testing robot used representations it learned using its own sensors. Condition 2: The testing robot's learned representation was combined with that of one other robot. Condition 3: The testing robot used a combined representations learned by *two* other robots. Condition 4: The testing robot used representations learned by *one* other robot.

used for training, and approximately fifty were used for testing (some objects had less than fifty).

Recognition of objects is performed using a feature-based vocabulary tree learned using the training instances (Nistér and H. Stewénius, 2006). We used a MATLAB implementation (Andrea Vedaldi, 2009). For each image, SIFT features and descriptors were calculated. As the training images were processed, the vocabulary tree was expanded based on similarity between the SIFT features of the training image and the current node in the tree (see Nistér and H. Stewénius, 2006 for details). We measured recall rates for all of the objects to determine classification accuracy.

Four different conditions were used (Figure 15). The first condition consisted of the same robot both learning and classifying ("Own Training"). This condition demonstrates accuracy when a robot learns using its own sensors. The second condition consisted of adding representations obtained from one other robot to this ("Combined Training – Incl. Own"). In other words, the robot learned using its own instances, but combined the resulting representation with that of another robot's. The third condition consisted of a robot classifying objects using representations received by the other two robots ("Combined Training – only Other"). In this case, no instances from the testing robot were used for training. Finally, the fourth condition consisted of a robot classifying objects using representations received from only one other robot ("Other Training"). All pairs of robots were used for these conditions.

Table 1 - Experimental summary for the experiment exploring the effect of heterogeneity on naïve knowledge transfer.

Experiment 1: General Experiment Summary **The Problem of Heterogeneity: A Motivating Experiment**	
Purpose	Demonstrate that perceptual heterogeneity can pose a problem during naïve knowledge transfer, even using modern computer vision techniques (vocabulary tree of SIFT features).
Experiment Type	Mobile Robots Amigobot and two Pioneer 2DX robots
Hypothesis	Heterogeneity will pose a problem during transfer of object models, i.e. lower average recall rates will be achieved when using object models learned by another robot as opposed to object models learned using the testing robot itself.
Procedure	**Training** 1) Fifty images per object are used to train vocabulary tree 2) Resulting vocabulary tree is used to classify test images
Independent Variable	Source of object model (represented as a vocabulary tree): Self, self & one other robot, two other robots, and one other robot
Dependent Variable	Average recall rate during classification of twelve objects.

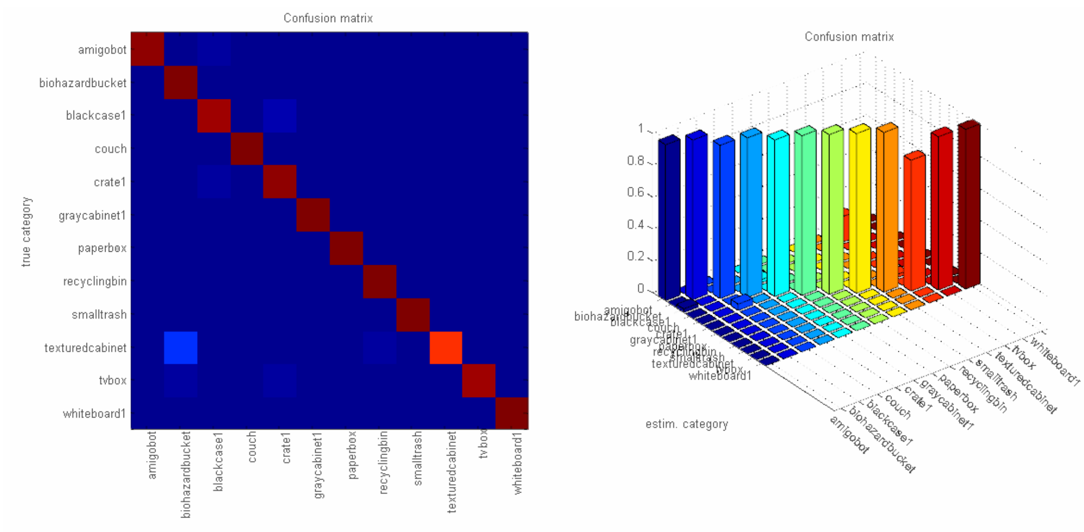


Figure 16 – Object classification results for the Amigobot, using representations it learned. Left: Confusion matrix depicting recall, where darker red colors represent higher values and darker blue colors represent lower values. Right: A bar chart showing the same confusion matrix. As can be seen, high values (greater than 0.9) are achieved for most objects. This is in constrast with the figure below depicting classification with representations obtained from another robot.

The hypothesis of the experiment is that learning with robot's own sensors will be better than transfer, even with features designed to be discriminative yet repeatable. Conditions where the robot's own learning is combined with other robots' learning is hypothesized to be next best. Only using other robots' representation is hypothesized to be worst. The basis for these hypotheses is that differences in the robots' sensing are anticipated to degrade transferred representations when processed by the receiving robot. Table 1 summarizes the experiment.

Figure 16 shows results for the Amigobot in the form of a confusion matrix, representing graphically the classification for each object. The actual object seen in the image is represented in each row, and each column shows what the robot classified the object as. High values in the diagonal, shown in red, represent good performance. In the graphics, higher values are represented as lighter blue and darker red, while lower values are shown as darker blue and lighter red. The right portion of the figure shows a 3D bar graph representation of the same data. As can be seen, classification accuracy for these objects is good. Figure 17 shows classification of the same objects by the Amigobot robot, this time using representations obtained from the Pioneer 2DX robot with a web camera. While many of the objects achieve high accuracies, the classification of a few of the objects fails catastrophically. This is due to differences in the features on the receiving robot, resulting from differences in the cameras or perspective.

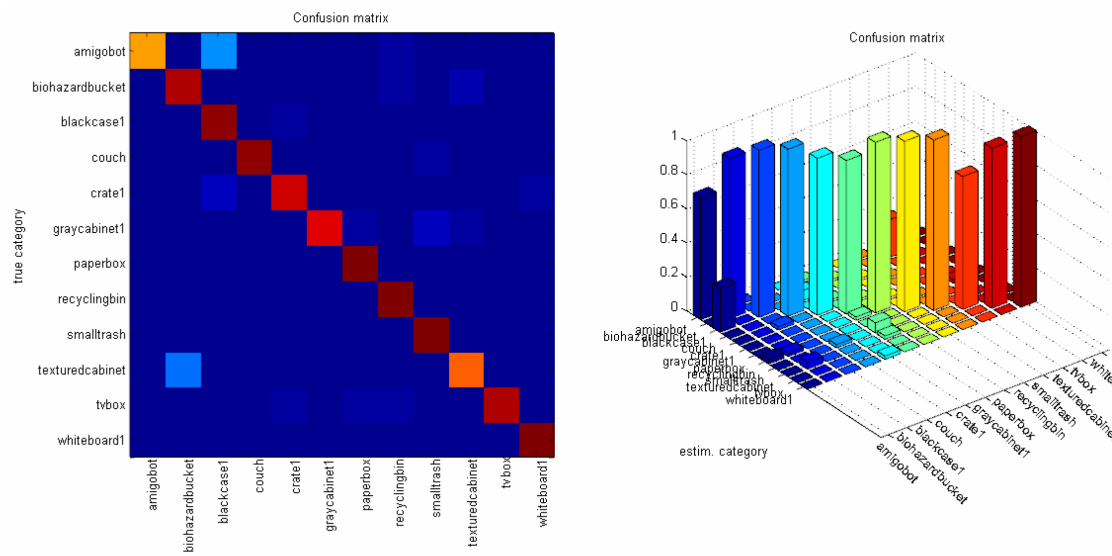


Figure 17 – Object classification results for the Amigobot, using representations it received from the Pioneer 2DX with web camera. Left: Confusion matrix depicting recall, where darker red colors represent higher values and darker blue colors represent lower values. Right: A bar chart showing the same confusion matrix. These results show catastrophic failures in the recognition rates (e.g. < 0.7) for two of the objects and smaller average recall rates over all objects.

Figure 18 shows a summary of these results as a graph, for all robots, in several conditions. The left column shows the recall results for the three robots, comparing accuracy resulting from using representations learned using the robot's own sensors versus using representations received from *each* of the other robots individually. An overall pattern is that learning with the testing robot's own sensors is clearly better than receiving representations from the other robots, if differences between the robots are not

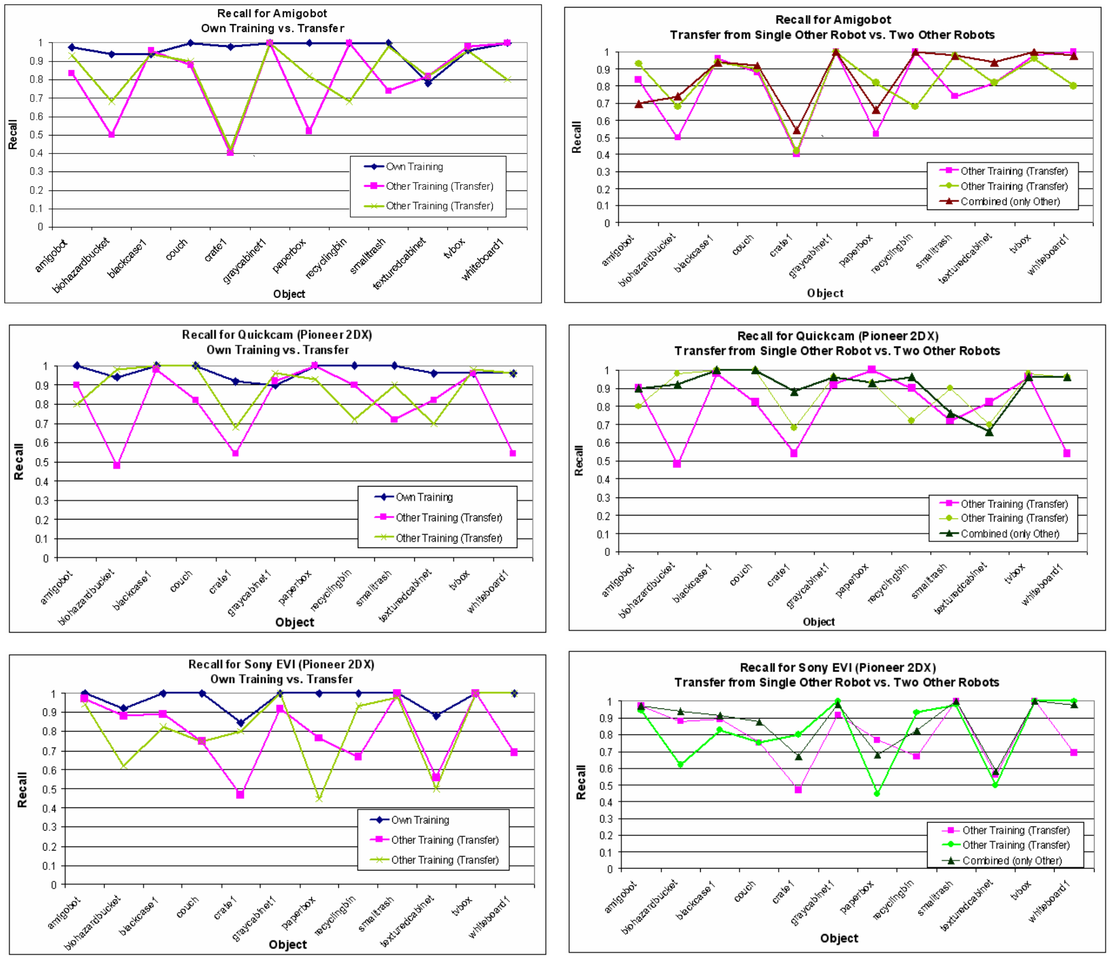


Figure 18 – Object classification results for the three robots and twelve objects in different conditions. Left: Recall rates are shown for all three robots when comparing models learned by the tested robot itself versus models received by one other robot. These figures show large dips in recall rates for certain objects, when compared to the "Own Training" condition. Right: Recall rates are shown for classification of objects given learned models obtained from one other robot ("Other Training (Transfer)") versus *two* other robots ("Combined (only Other)"). Combining learned models from other robots resulted in higher average recall.

analyzed. While some objects transfer well, there are several catastrophic failures where the accuracy drops to extremely low levels. This is a consistent pattern across all combinations of robots.

The right column compares accuracy resulting using representations received from *each* of the other robots (individually) versus using representations received from *both* of the other robots (together). Interestingly, if transfer is to occur from other sources,

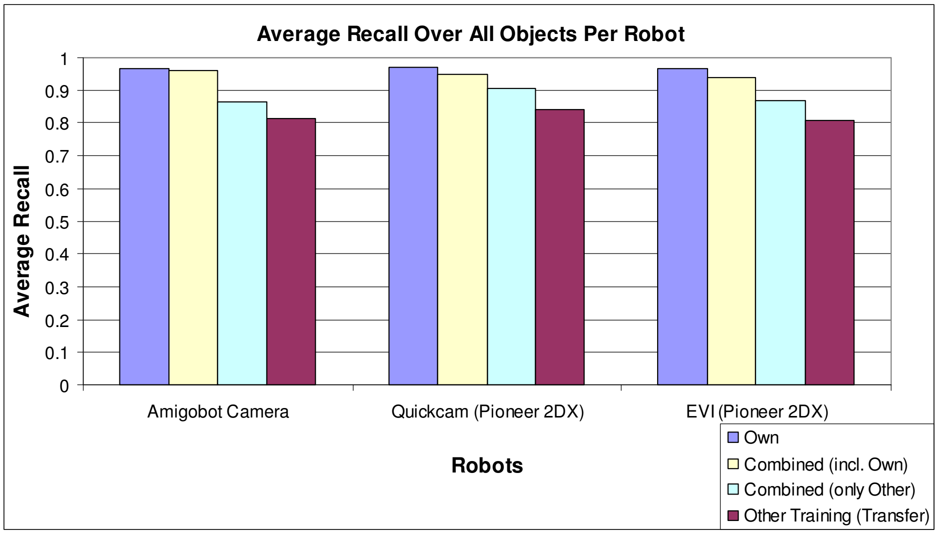


Figure 19 – Object classification results for the three robots, averaged over all objects, in the four conditions. These results demonstrate that classification by one robot using representations obtained from other robots ("Combined (only Other)" and "Other Training (Transfer)" conditions) are less effective than ones learned by the tested robot itself. The graph also shows that the results are consistent across the three robots.

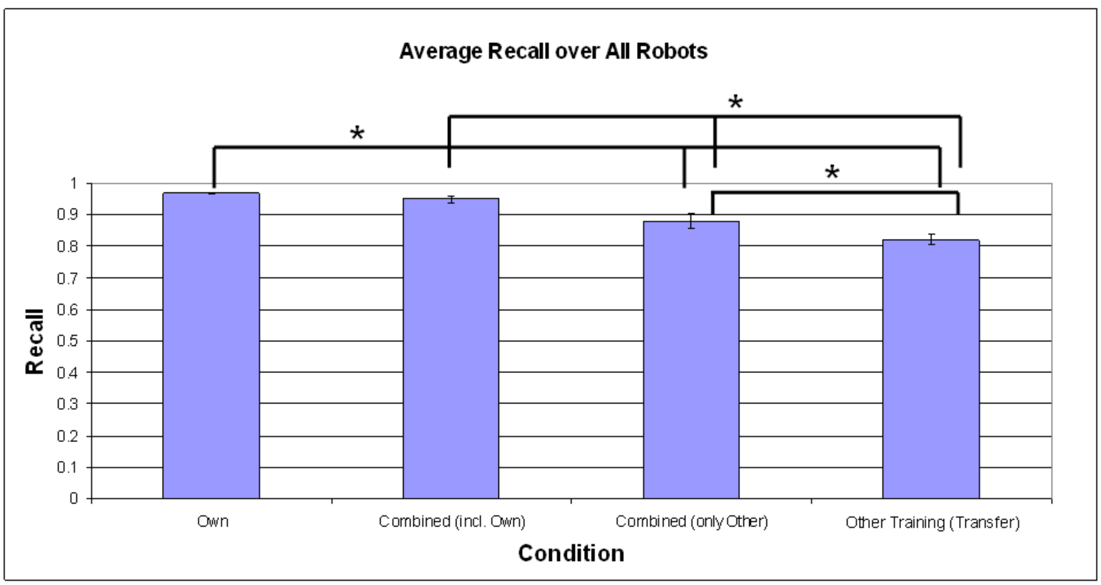


Figure 20 - Object classification results averaged over all three robots, in the four conditions. Starred combinations represents significant differences. These results demonstrate that classification by one robot using representations obtained from other robots ("Combined (only Other)" and "Other Training (Transfer)" conditions) are less effective than ones learned by the robot itself or obtained from one other robot but combined with its own learning ("Own" and "Combined (incl. Own)").

receiving learned representations from two other robots is better (averaged over all objects) than from just one. In this case, heterogeneity is beneficial because it increases the probability that features that the receiving robot can detect will be transferred.

Table 2 - Experimental summary and conlusions for the experiment exploring the effect of heterogeneity on naïve knowledge transfer.

Experiment 1: General Experiment Conclusions **The Problem of Heterogeneity: A Motivating Experiment**	
Hypothesis	Heterogeneity will pose a problem during transfer of object models, i.e. lower average recall rates will be achieved when using object models learned by another robot as opposed to object models learned using the testing robot itself.
Conclusions	Hypothesis is supported. Direct transfer of object models resulted in catastrophic failures and lower average recalls.

Figure 19 summarizes the results, averaging the recall rate over all objects for the three robots in the four conditions. Figure 20 shows the same results averaged over all three robots with standard deviation (significant results are starred). As can be seen, the highest accuracy is achieved when learning with the robot's own sensors. Receiving representations learned from another robot, when it is added to learning with the robot's own sensors, is slightly worse than learning with just the robot's own representation, although more data is needed to clarify this relationship as the difference from the first condition is not significant. Third best is receiving representations learned by *two* other robots. Receiving representations from only one other robot achieves the lowest average. Although the degradations in averages are not extremely large, they represent catastrophic (i.e. recall rates < 0.7) failures in individual objects not graceful degradation across all objects. This could be seen from the detailed graphs in Figure 18. Note that the patterns are consistent for all three robots. Table 2 summarizes the conclusions for the experiment.

Clearly, there is an advantage to each robot learning by experiencing the world with its own sensors. The lesson here is not that the transfer of learned knowledge from other robots *can't* be done; clearly, for many objects it was effective. Instead, transfer cannot occur blindly. Before transferring knowledge, the robots must understand what differences exist between them, how these differences will affect transfer, and which particular pieces of knowledge would or would not transfer well. This is true even when using modern computer vision algorithms that use repeatable features designed to be utilized over images with different types of variations in lighting, scale, or other transformations.

Another conclusion that can be drawn from this experiment is that learning with one's own sensing is important, and hence if transfer is used to bootstrap learning, the representation should support continued learning by the receiving robot. Allowing robots to learn explicit models of their differences in order to facilitate transfer, predict when it will fail, and continue learning after transfer is the topic of this dissertation. *Instead of using representations that only work for vision, however, we use a more general multi-level representation that can represent concepts with many features that come from different modalities***.** This representation is described in the rest of this chapter, beginning with the next section.

3.2. Representation Overview

In order to answer our research questions, we must commit to a representation of the world, including a description of features and objects. While the representation used in the experiment above works with images, we are interested in higher-level *concepts*, which can describe objects using many types of object characteristics obtained through multiple modalities. We use a grounded symbolic feature-based representation to describe the general characteristics of the world. The representation construction begins with low-level sensors and features, and then abstracts these low-level observations into properties and concepts (which can represent object categories and specific objects). In our earlier example, a CCD video sensor can provide images in the form of red, green, and blue values for each pixel. A perceptual feature detector can then process this image to produce a set of color features (such as average RGB values or histograms) for salient regions of the image. *Properties* can be derived by specifying regions in these multi-dimensional perceptual features; for example, 'green' can correspond to a specific region in the RGB color space.

Objects can have real-valued memberships in multiple properties (e.g. an apple can be various shades of green or red), allowing for a richer representation and the ability to handle uncertainty in sensing. Also, regions in different spaces can result in similar properties of objects; for example, similar color property can be learned using both HSV and RGB color spaces. Object categories and specific objects are defined by combining multiple feature categories together (e.g. "small", "green", and "round") (Gärdenfors, 2000). Object categories are more general than specific objects, which can have location and more specialized properties. For example, a specific apple may be in a certain location in the environment, a specific color, or have a specific texture. Object categories are hence more general, and there can be many specific instances of them (just as there are many types of apples). Finally, object categories may be nested hierarchically, so that apples and oranges can be grouped into "fruits". In search and rescue, concepts such as humans and animals can be grouped into "living", for example in order to alert an operator that there may be survivors. Hence, there are multiple layers of representation, ranging from simple features such as specific colors to groupings of object types. The specific representation and computations necessary are described in the next subsection.

There are several benefits to this multi-level representation. First, we claim and will provide evidence (in this chapter and Chapter 5) that the property abstraction aids both individual learning and transfer between robots. Another advantage of property abstraction involves the amount of effort needed to model robot differences. The intuition is that a small number of properties can be used to represent an order-of-magnitude larger number of concepts. Since there will be only a few number of properties to be mapped, compared to the number of concepts, less effort is needed for transfer to occur. Note that this abstraction also allows the robots to represent the same properties using their own sensing, and can even use different underlying spaces (for example, an HSV color space versus an RGB color space) or different modalities (for example, the width of an object can be sensed via laser or stereo camera). We will provide evidence for the claim that this aids both learning as well as transfer, especially when the underlying representations used by the robots differ, in the experimental section of this chapter as well as Chapter 5.

A second major benefit of such a hierarchical representation is that structured knowledge can be shared among the robots depending on the level at which they significantly diverge. If robots share many properties and concepts then entire ontologies based on these can be communicated with very little interaction. This requires that they know what their differences are at multiple levels; the process of building and maintaining such models is a major topic that is covered in this thesis. For example, the two robots in our example may have similar properties for color (since they share a CCD camera and color features), and if they do one robot can forward its knowledge regarding concepts represented as combinations of these properties to the other. If the underlying properties are not completely shared, then the concept representation must be modified or taught to the receiving robot in other ways. For example, robots can lead each other to similar points of view and trade labels for concepts in the world, where each robot must then learn its own respective grounded representation.

In other words, the richness of communication possible depends on what is shared and what is not. In the upcoming sections, we will detail our representation, and the different possibilities involving knowledge transfer are explored in detail in Chapter 5.

3.3. Sensors, Features, Properties, and Concepts

We will now describe the representation formally. Note that Figure 21 summarizes an example representation for a more intuitive depiction. Each robot has a set of m sensors $S = \{s_1, s_2, ..., s_m\}$. We denote the number of sensors as $|S|$. As seen from the example, these can include CCD camera or laser. At time t, the robot receives an observation vector $o_{t,i}$ from each sensor s_i, resulting in a set of measurement or observation vectors $O_t = \{o_{t,1}, o_{t,2}, ..., o_{t,|S|}\}$ (in certain cases we may omit the subscript for simplicity). For example, a SICK laser sensor can provide a vector of 180 values at each time instance. The preceding notation refers to the sensors and observations (and later features, properties, and concepts) of a single robot. We denote the robots with a superscript, so that s_i^j is sensor i of robot j. This is true for notation used in subsequent sections (for example, for features). For clarity, the subscripts are omitted when describing a general robot in our framework.

Sensor data provide a stream of unprocessed information so it is presumed that each

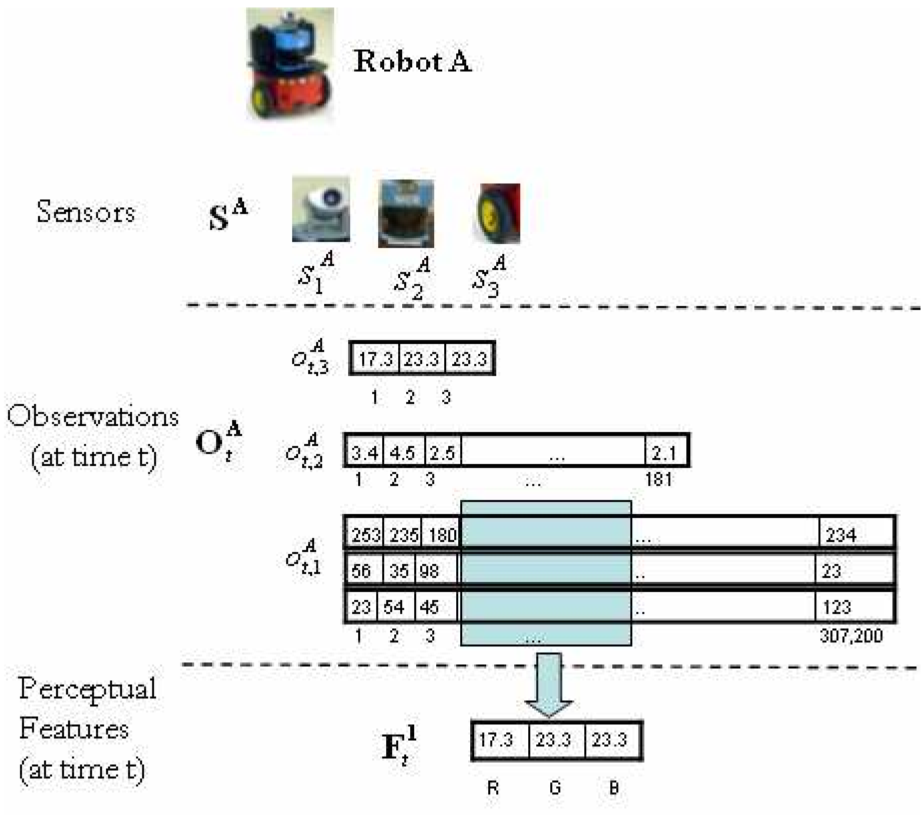


Figure 21 – Example depiction of sensors, observations, and perceptual features. Sensors obtain data of physical properties in the world, creating observations at every time instant. Observations are then processed to produce perceptual features, functions of salient segments of observation data.

robot has a set of p feature detectors, $F = \{f_1, f_2, ..., f_p\}$, that further process observations and output perceptual features. We denote specific values of a set of features at time t as $\mathbf{F}_t$, and the specific value of a feature i as $f_{t,i}$. A feature detector is a function φ that maps a set of observation vectors into a set of feature vectors, i.e. $f_i = \Phi(\mathbf{O}_{f_i})$ where $\mathbf{O}_{f_i} \subseteq \mathbf{O}$ denotes the set of input observations used by the feature detector. As an example, a perceptual feature for robot A in the example can be an average HSV color of salient regions in an image. There may be multiple such regions, and hence a set of vectors may be returned. Another example would be a blob detector that takes a camera image as input and outputs a vector specifying a list of blobs found and their positions. Figure 21 depicts sensors, observations, and perceptual features for example robot A.

The observation and feature vectors contain data received by the robot from the world. We use conceptual spaces (Gärdenfors, 2000) to anchor this data to learned concepts. We combine the original formulation proposed by Gärdenfors and some of the subsequent generalizations and extensions (Rickard, 2006). Conceptual spaces are geometric spaces that utilize similarity measures as the basis of categorization. This geometric form of representation has several advantages. It is amenable to measurement of uncertainty (Rickard, 2006) and various machine learning algorithms such as clustering that operate in similar spaces. It has also been elaborated upon and extended in several other works (Aisbett and Gibbon, 2001),(Raubal, 2004),(Rickard, 2006), and discussed and implemented to a limited degree in robotic systems (Balkenius et al., 2000),(Chella et al., 2004),(LeBlanc and Saffiotti, 2007). Most importantly, understanding how different properties and concepts can be mapped between different agents can be intuitively viewed in these spaces. As an added benefit, it has been used to understand and more concretely define various categorization theories, e.g. those raised in psychology (Rosch, 1988). As a result of these roots in psychology, several issues such as the combinations of categories in language where properties may overtake one another (e.g. a "small elephant") or context-dependent categorization (e.g. "red wine", which is not the typical "red" used in other contexts) have been addressed.

The most basic primitive of the representation is a *dimension* (also referred to as *quality* or *attribute*), which takes values from a specific range of possible values (a domain in the mathematical sense, although it is not to be confused with the notion of a domain used in the next paragraph). For example, the *hue* of an object can be specified as an angle in the range [0, 360]. The values of these dimensions come from perceptual features described above. For example, a video sensor measures physical properties of the world (light), converting them into a digital representation consisting of multiple pixels in the form of RGB space. A perceptual feature detector can then convert regions of the image into an HSV space, and the H (hue) value can make up a dimension. The feature detector returns a set of these, one for each region of the image that it determines is salient. The left side of Figure 22 depicts this process.

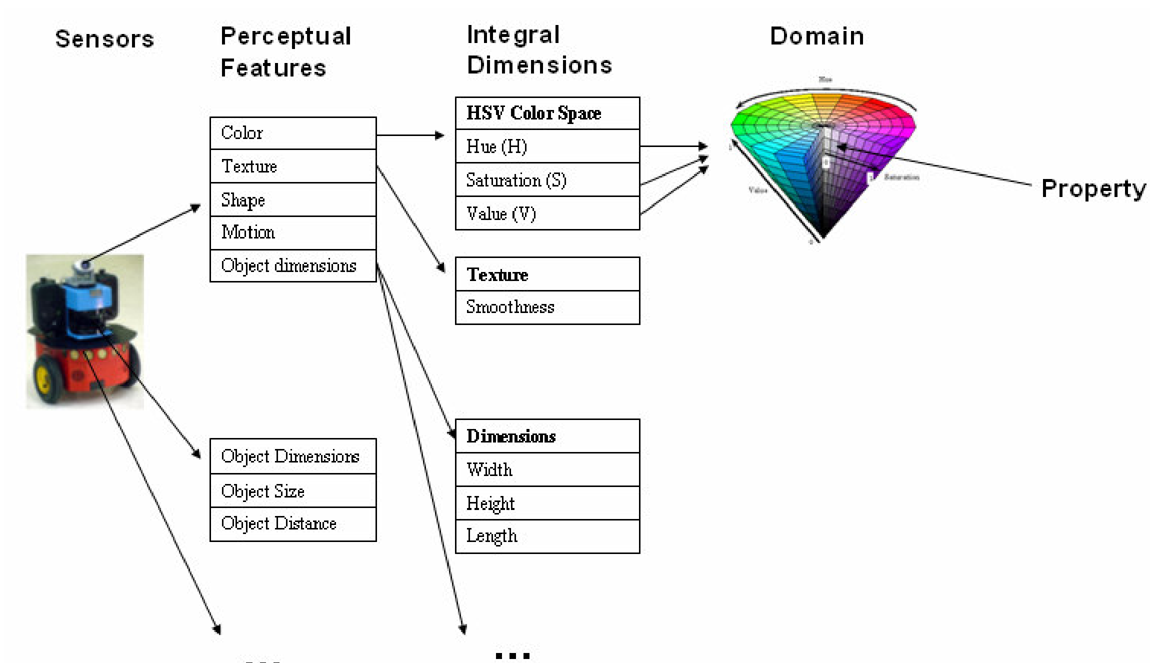


Figure 22 - Example of processing for sensors, perceptual features, and conversion to integral dimensions, domains, and properties for example robot A.

Gärdenfors posits that there are *integral* dimensions that cannot be separated in a perceptual sense. For example, the HSV color space can be argued to consist of three integral dimensions. Another example used is pitch and volume that is perceived by the auditory system. A set of such integral dimensions is referred to as a *domain*. A domain defines a space that consists of all possible values of the integral dimensions. It is useful to abstract and divide these values into specific regions, which define a *property*. For example, "blue" can be a property that corresponds to some region of the color space.

The regions can be arbitrary shapes, although Gärdenfors defines what he calls *natural properties* consisting of regions with certain characteristics such as convexity. Note that a property corresponds to a region in a single domain. Figure 22 shows the various stages of processing for robot A in our working example.

We can now define a *conceptual space* K as made up of a set of domains. A specific *concept* in the conceptual space is a set of regions from the domains $D = \{d_1, d_2, ..., d_n\}$ in the conceptual space (Gärdenfors, 2000). A concept may also contain *salience weights* for properties and *correlations* between the properties. For some concepts, certain properties can be more important than others, and this can be influenced by task context as well; for example, in the context of eating, whether the color of an apple is green or red may not matter much. Correlations between properties for a concept can also exist, for example if the color of an apple correlates with its texture (e.g. a brown color can be correlated with being wrinkled if the apple is rotten).

A point in the conceptual space is called a *knoxel* $k = < k_1, k_2, ..., k_n >$, and specifies instances of the concept in the form of vectors. A knoxel can specify points in some of the domains, while leaving others unspecified, in the form of a partial vector. Note that a property is a specific type of concept that utilizes only one of the domains from the conceptual space.

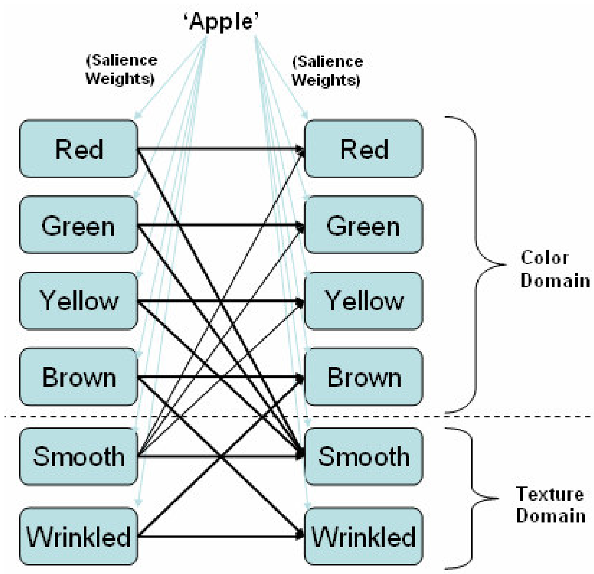
Table 3 - Properties for robot A and B in our example. The table headings give intuitive labels for the reader. Below that, the notation for properties (e.g. p_1^A) is shown as well as a random symbol to depict the notion that robots cannot simply compare symbols when learning about their similarities and differences with respect to these properties.

Brown	Black	Blue	Gray	White	Red	Green	Yellow	Wrinkled	Smooth
p_1^A	p_2^A	p_3^A	p_4^A	p_5^A	p_6^A	p_7^A	p_8^A	p_9^A	p_{10}^A
'p_A_1289'	'p_A_2345'	'p_A_8723'	'p_A_1287'	'p_A_9875'	'p_A_6423'	'p_A_1826'	'p_A_1756'	'p_A_98'	'p_A_62'
p_1^B	p_3^B	p_5^B	p_2^B	p_4^B	p_8^B	p_7^B	p_6^B	None	None
'p_B_8752'	'p_B_5428'	'p_B_1342'	'p_B_9877'	'p_B_1984'	'p_B_7831'	'p_B_1756'	'p_B_1876'	'p_B_42'	'p_B_58'

In order to abstract features and to facilitate communication, symbols are attached to properties and concepts. Each robot maintains a set of symbols X, each of which is grounded to a concept via the representation. Symbols correspond to labels or strings, which will be randomly assigned by the robot. A representation can be described as a function that returns the degree to which a specific knoxel k can be categorized as a concept represented by symbol $x \in \mathrm{X}$; i.e. $R:(k\ ,x) \rightarrow [0,1]$. A subset $P \subseteq \mathrm{X}$ of these symbols are predicate symbols (e.g. 'blue') which are grounded to property concepts. Table 3 shows property numbers and symbols in our example; note that each robot will have different groundings for these properties. The mappings between the property numbers and symbols in each robot are *not* known *a priori*, and is part of the information that is autonomously learned via joint exploration and the algorithms outlined in this dissertation. Property numbers are assigned randomly for robot B in order to avoid bias in the algorithms used later on to infer these mappings. Each property concept $p \in P$ has a *prototype* for that property in the form of a knoxel, denoted as k_p. This can be calculated, for example, by finding the center of mass for the region defining the property. Note that all symbols are grounded to perceptual features derived from sensing.

There are several ways in which properties, with associated weights and correlations, can be combined to represent a concept. Often, properties are represented as well-defined regions in domains, and when an instance is in one of these regions, it is said to have the corresponding property (Chella et al., 2004). In other words, instances or objects are defined via a conjunction of predicates (corresponding to properties), where the level of membership in the property is lost. For example, an instance may be on the edge of the region defined by the property (e.g. only slightly tall), but this information is not retained. Obviously, this type of discretization presents issues regarding uncertainty. In our work, we will take into account the degree of membership for a property, as well as the degree of membership for a concept, allowing the robot to take this in consideration during the communication process and when deciding how to act.

In order to do this, we used the extension of conceptual spaces to allow fuzzy memberships proposed by Rickard (Rickard, 2006). A concept is represented as a graph of nodes consisting of properties, with salience weights for the concept. Nodes for pairs of properties p_i and p_j are connected with directional edges, with weight $C_{(i,j)}$, corresponding to the conditional probability that the concept will have property p_j given that it has property p_i (Rickard, 2006). If the two properties are disjoint (non-overlapping regions) and are from the same domain, $C_{(i,j)} = 0$. The graph can be represented as a non-symmetric square connection matrix. The concept graph for an example "apple" concept can be seen on the left side of Figure 23 (with edge weights represented by arrow thickness), and the resulting matrix on the right side of Figure 23.



$$
\begin{array}{r|cccccc|}
 & \text{Red} & \text{Green} & \text{Yellow} & \text{Brown} & \text{Smooth} & \text{Wrinkled} \\
\text{Red} & 1 & 0 & 0 & 0 & 1 & 0 \\
\text{Green} & 0 & 1 & 0 & 0 & 1 & 0 \\
\text{Yellow} & 0 & 0 & 1 & 0 & 1 & 0 \\
\text{Brown} & 0 & 0 & 0 & 1 & 0 & 1 \\
\text{Smooth} & 0.5 & 0.2 & 0.3 & 0 & 1 & 0 \\
\text{Wrinkled} & 0 & 0 & 0 & 1 & 0 & 1
\end{array}
$$

Figure 23 – Left: Example concept graph for 'apple' concept. Right: Matrix representation for an example "apple" concept with six properties (Rickard, 2006).

3.4. Calculating Concept Memberships and Concept Similarity

In order to compare concepts and categorize instances, the concept matrix described above can be projected into a hypercube graph representation (Rickard, 2006). Specifically, the matrix C is converted into a vector c by concatenating subsequent rows together, so that values from row i and column j correspond to element $r = (i-1)N + j$ in the vector, where N is the dimensionality of the matrix (corresponding to the number of properties). A depiction of this conversion can be seen in Figure 24. The salience weights of the properties can be multiplied so that $w_r = w_i \bullet w_j$. Concept similarity between two concept vectors c and c' can now be defined as the fuzzy mutual subsethood, a similarity measure for fuzzy sets (Rickard, 2006):

$$s(c,c') = \frac{\sum_r w_r \min(c_r, c'_r)}{\sum_r w_r \max(c_r, c'_r)} \qquad \textbf{(1)}$$

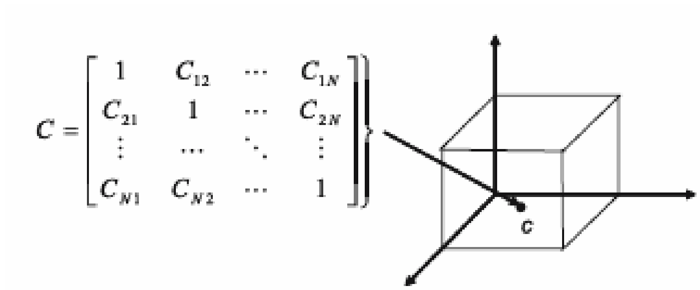


Figure 24 - Depiction of transformation from concept matrix representation to a hypercube (Rickard, 2006). The left side shows the concept matrix while the right side depicts the fact that the concept matrix is unrolled and corresponds to a point in the concept space.

The concepts may not share all properties, in which case the row and column corresponding to the property can be zeroed out, removing its contribution. We will use this property later as well, when concepts are shared between different robots which may not share all of the same properties.

In order to categorize the degree to which a specific *instance* belongs to a concept, a similar method is used. Table 4 summarizes the algorithm. The instance is converted to a graph (represented as a connection matrix as well) and compared to the concept matrix. First, the instance must be converted into the hypercube representation used for concepts. Let D_c be the domains in the conceptual space and P_c be the set of all properties that are involved in the target concept, i.e., a property that has a nonzero connection between it and another property (or vice-versa). This can be defined as:

$$P_c = \{ p \in P : \exists j \text{ s.t.} C_{pj} > 0 \text{ or } C_{jp} > 0 \} \quad \textbf{(2)}$$

Table 4 - Algorithm for obtaining a concept membership value given an instance and a concept.

Algorithm: Concept Memberships of an Instance

Input: Instance *i*, Domains *D*, Properties *P*, Concept *c*

Output: Property membership *m*

// Find all zero and non-zero values in the concept matrix

I_c = Nonzero members of concept *c* **(equation 2)**

I_{cn} = Zero members of concept *c*

DomainMax = *n* x *n* matrix initialized with zeros, where *n* = number of domain

```
// Find maximal value for diagonal of matrix DomainMax
For each Domain
   Find property MaxProperty such that it is the maximal property value in domain
   DomainMax[domain][domain] = i.propertyValue(MaxProperty)
End

// Find maximal value for non-diagonal of matrix DomainMax
For each Property P1
   For each Property P2
      MinValue = min( i.propertyValue(P1), i.propertyValue(P2) )
      If  (MinValue > DomainMax[ P1.domain ][ P2.domain ])
         DomainMax[ P1.domain ][ P2.domain ] = MinValue
    End
End

// Now we can fill in actual instance matrix MI
For each Property P1
   For each Property P2
     // For the diagonal of the matrix, set to actual property membership value
     If ( P1 == P2 )
       MI[P1][P2] = i.propertyValue(P1);
     Else
       // Fill in maximum across the corresponding domains
       MI[P1][P2] = DomainMax[ P1.domain ][ P2.domain ]
     Endif
   End
End
```

// Clear values where the concept matrix entries are zero

MI(I_{cn}) = 0

PropertyMembership = Calculate mutual subsethood between *c* and MI **(equation 3)**

The matrix I_c is then defined as:

$$I_{c,(i,j)} = \begin{cases} \max\limits_{j \in D_c, k \in D_c} \min\limits_{j \in P_c, k \in P_c} (s(i, p_j), s(i, p_k)) & \text{for all i, j} \in P_c \\ 0 & \text{otherwise} \end{cases} \qquad \textbf{(3)}$$

where $s(i,p_j)$ represents the membership (similarity) of an instance to a property, which is derived from the regions representing the property (described in the next subsection). These formulations are derived from fuzzy set theory, and their justifications are elaborated upon in (Rickard, 2006). They have also been successfully used in image matching tasks, for example (Ionescu and Ralescu, 2006). An example matrix from an "apple" instance, along with the general "apple" concept matrix is depicted in Figure 25.

"Apple" concept graph C:

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 0.5 & 0.2 & 0.3 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 \end{bmatrix}$$

Object graph Ψ:

$$\begin{array}{r c} & \begin{array}{cccccc} \text{Red} & \text{Green} & \text{Yellow} & \text{Brown} & \text{Smooth} & \text{Wrinkled} \end{array} \\ \begin{array}{r} \text{Red} \\ \text{Green} \\ \text{Yellow} \\ \text{Brown} \\ \text{Smooth} \\ \text{Wrinkled} \end{array} & \begin{bmatrix} 0.9 & 0 & 0 & 0 & 0.7 & 0 \\ 0 & 0.9 & 0 & 0 & 0.7 & 0 \\ 0 & 0 & 0.9 & 0 & 0.7 & 0 \\ 0 & 0 & 0 & 0.9 & 0 & 0.7 \\ 0.7 & 0.7 & 0.7 & 0 & 0.7 & 0 \\ 0 & 0 & 0 & 0.7 & 0 & 0.7 \end{bmatrix} \end{array}$$

Figure 25 - Depicts the concept graph of an "apple" (left) versus a specific instance transformed into the same matrix representation (right) (Rickard, 2006).

Given this matrix, the membership of an instance to a concept can be defined using the fuzzy mutual subsethood equation:

$$s(c,i) = \frac{\sum\limits_{i \in P_C} w_i \times \min(c_i, I_c)}{\sum\limits_{i \in P_C} w_i \times \max(c_i, I_c)} \qquad \textbf{(4)}$$

where w_i is the property weight. In the example, the calculations are performed as follows:

$$s(c,i) = \frac{4*\min(0.9,1) + 7*\min(0.7,1) + \min(0.5,0.7) + \min(0.2,0.7) + \min(0.3,0.7)}{11*1 + 0.5 + 0.2 + 0.3}$$
$$= \frac{4*0.9 + 7*0.7 + 0.5 + 0.2 + 0.3}{11*1 + 0.5 + 0.2 + 0.3} = 0.7917$$

The resulting value represents a membership of instance i to concept c, where higher values indicate that the instance is more similar to the concept. Note that here we diverge from (Rickard, 2006) which uses fuzzy subsethood instead of mutual subsethood. We have empirically found better performance with the latter method.

3.5. Learning Properties and Concepts from Observation

In order to learn a representation for objects, we will manually scaffold the robot's learning by first providing it with property labels and instances, after which concept instances are provided. Properties represent regions in a domain, and can be learned via supervised learning where specific symbols (or labels) are given based on the sensory data of the robot. The scaffolding is provided using experimenter intuition.

Each scene, which can contain multiple properties and concepts, results in a set of knoxels K calculated from the output of the robot's perceptual feature detectors. A supervised symbol is associated with each scene. For each property, we use a Gaussian Mixture Model (GMM) to characterize the regions, denoted as G_i for property p_i. In this dissertation, we used one property per scene. Future work can investigate multiple properties. If there are multiple properties per scene, the clusters can be built for all domains, and the correct cluster can be inferred by its frequency of being selected. In other words, models will be built for all domains, and as they become more accurate incorrect domains will be ruled out because new instances will not correspond to the same clusters (properties) as previous instances.

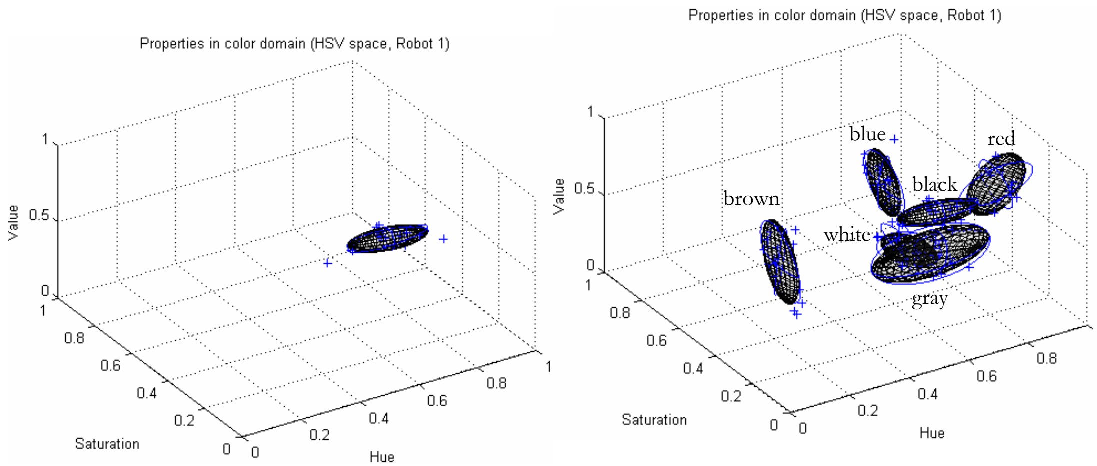


Figure 26 – Properties, represented as Gaussian clusters, in the color domain (HSV color space). Left: Property corresponding to black color. Right: Several properties in the same domain, corresponding to colors blue, black, gray, brown, red, and white.

Specifically, *each* property can be modeled as:

$$P(k \mid \theta) = \sum_{i=1}^{k} w_j P(k \mid \mu_j, \Sigma_j) \qquad \textbf{(5)}$$

where w_j is known as the mixing proportions and θ is a set containing all of the mixing proportions and model parameters (mean μ and standard deviation Σ). Figure 26 shows one possible clustering for both a single color property (black) on the left as well as a collection of properties in the same color domain (blue, black, brown, gray, red, and white) on the right. The ellipsoids represent the Gaussian clusters, which in this case was limited to one per property.

Since each property is modeled as a mixture of Gaussians, where it is not known which Gaussian the data came from, a data association problem must be solved. There will be several clusters in the space, and the algorithm must first determine which cluster the data belongs to before updating the parameters of the model. One method to solve this, which we will use, is Expectation Maximization, which alternates between estimating the association of the points to the clusters and updating the parameters of the clusters given the association (Bilmes, 1998). In this thesis, we typically use one Gaussian per property in order to maintain the convexity of a property. Table 5 summarizes the algorithm for learning a property from instance.

Table 5 - Algorithm for learning a property from data.

Algorithm: Learning a Property from Data

```
Input: Feature detectors F, Domain d
Output: Property P

// Gather all the data
For each instance i
   // Calculate feature vector (e.g. RGB) but only for target domain (e.g. color
domain)
   For each f in F
      If (Domain(f) == d)
        // This feature detector is in the right domain
        Values.add( f(i) )
      End
   End

   TrainingSet.add(Values)
End

P = ExpectationMaximization(TrainingSet)
Return P
```

Once the properties have stabilized (in that their associations and parameters do not change greatly), concepts can also be learned via supervised learning (Rickard, 2006). Instances again take the form of sensory readings and a corresponding label, this time to a concept. The feature vectors are processed from sensory data and then placed as dimensions in their respective domains, and property memberships are calculated using a similarity metric (for example, weighted Euclidean distance from the centroid of the region). Specifically, given a set of instances processed into knoxels K_i, the instances are converted into a matrix I where $I_{i,j}$ contains the similarity between the property membership for property p_j in instance *i* and the prototype of property p_j (denoted as k_{p_j}). Membership of a property (similarity between the property and its prototype) is measured using the metric defined in Section 3.4. Each element of the concept matrix described previously is then updated in an incremental manner, determined by a learning rate α, as follows:

$$C_{(j,k)} = C_{(j,k)} + \left(\frac{\min(I_{i,j}, I_{i,k})}{I_{i,j}} - C_{(j,k)}\right) \times \alpha \qquad \forall j \neq k \tag{6}$$

For each instance, this equation calculates the ratio of membership to both properties divided by their membership in the first property. Here, membership in both properties is represented as the minimum of the two. In the case of when *j=k*, this membership ratio in equation (6) always equals one. However, we wish to retain the actual strength of the membership instead. Hence, we increment the value towards the actual membership in property *j* multiplied by the learning rate. This diverges from the work in (Rickard, 2006) but resulted in better empirical performance.

The cell value corresponding to those two properties is then moved toward this result, modulated by a learning rate α. Learning rates are typically used to achieve good noise

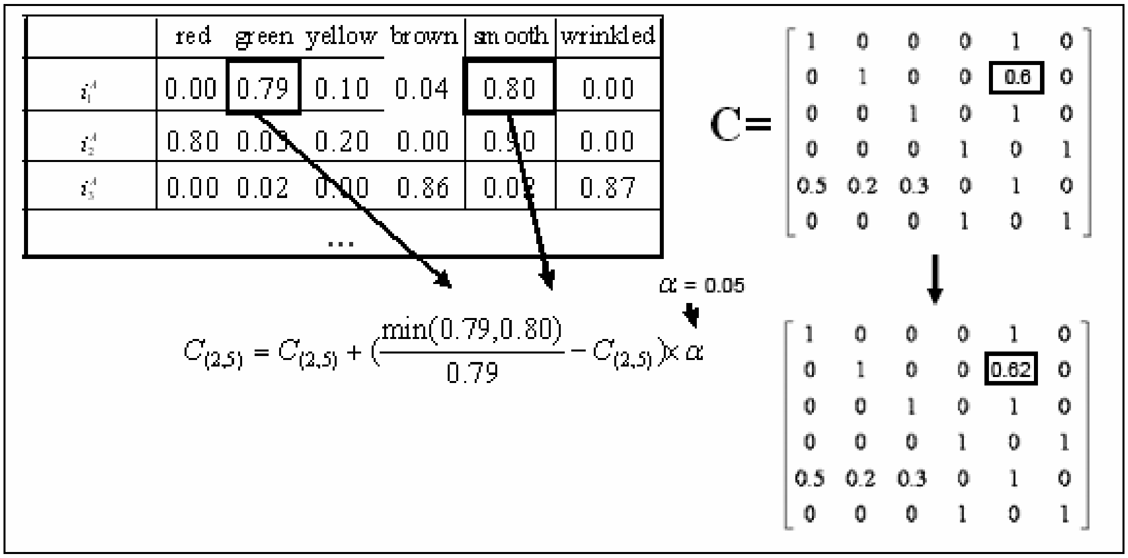


Figure 27 –Example demonstrating the update of one cell of the concept matrix, based on one instance. This is done for all instances that are members of this concept.

characteristics (since outliers will not affect the values drastically) and also makes the calculations incremental. Learning that is incremental is important, since after knowledge transfer the receiving robot has to continue learning using its own sensors if it encounters instances itself. Figure 27 shows an example for the update of one cell in this matrix. Table 6 summarizes the algorithm.

3.6. Defining Perceptual Heterogeneity

Using conceptual spaces as a representation for objects and their properties, we can now define the types of perceptual heterogeneity that we will be dealing with. One way to look at the issue is to identify the underlying sources of heterogeneity that led to heterogeneity at the conceptual level. In robots, differences in sensors, perceptual features, and experiences (e.g. instances that are used for learning) can all cause heterogeneity at the conceptual level. Given these underlying causes of conceptual heterogeneity, it becomes clear that in a practical sense all robots will have at least some degree of heterogeneity. For example, even sensors that are of the same model will differ somewhat; this problem has been encountered by related work that attempted to implement the Talking Heads experiment on two Aibo robots (Nottale and Baillie, 2007). Similarly, the experiences of two robots will likely not be exactly the same, especially in the presence of sensor noise.

Table 6 – Algorithm for learning a concept from data.

Algorithm: Learning a Concept from Data

```
Input: Instances I , Learning Rate α
Output: Concept Matrix C

For each instance i
  For each property P1
    For each property P2

      // If it is the same property, increment towards the membership value
      If  P1 == P2
        Value = i(P1)
      Else
        // Get minimum value
        MinValue = min( i(P1), i(P2) )

        // Divide by the first property membership
        Value = MinValue / i(P1)
      End

      C[P1][P2] = C[P1][P2] + (Value - C[P1][P2]) * α

    End
  End
End

Return C
```

Although all robots can be argued to be heterogeneous in some sense, the propagation of this heterogeneity to the conceptual level is what matters in the end. If two robots differ somewhat in sensing but yield the same concept to an acceptable degree of certainty, then communication becomes possible. However, the problem is that since the same concept may be grounded differently in the two robots; which concepts are mapped to each other must be determined. Furthermore, even the dimensions used to define the concepts may differ between the two robots.

We will now define several classes of heterogeneity at the level of properties and concepts, analyzing the differences in the different levels of the representation (Figure 28).

H1: Differences in Domains

H1a: All integral dimensions are shared.

H1b: Some dimensions are shared.

H1c: One or more quality dimensions from robot A combinesa set of quality dimensions from robot B together.

Examples of this occur in children, for example confusing notions of volume with height alone, or mass and weight (Gärdenfors, 2004).

H1d: Domain exists in robot A but does not exist in robot B.

Note: In addition, there may be differences in the types of sensors used to obtain values for the same dimension (e.g. object size via camera and laser), or even differences in the parameters of the sensors (e.g. two camera with different color characteristics). These lower-level differences may or may not translate into differences in dimensions, properties, and concepts, which is what we are ultimately interested in.

H2: Differences in Properties

H2a: Regions in shared domain are equal (i.e. regions are completely overlapping).

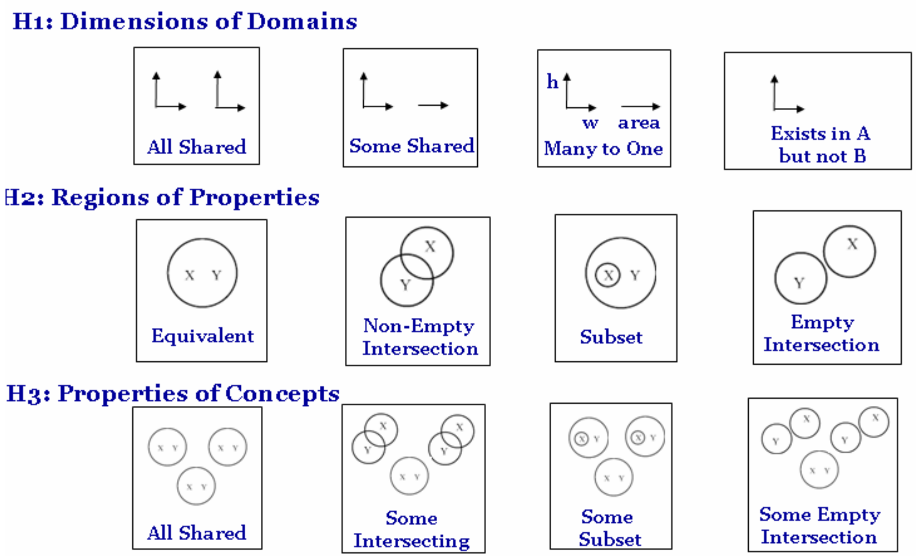


Figure 28 – Classes of heterogeneity defined by differences in multiple levels of representation.

H2b: The intersection of some regions in a shared domain is not empty.

H2c: One region is a complete subset of the other (i.e. the intersection of two regions is equal to one of the regions).

H2d: The intersection of regions in shared domain is empty.

H3: Differences in Concepts

H3a: All properties are shared.

H3b: Overlapping sets of properties.

H3c: One or more property/concept is subset of other properties.

H3d: All or some properties are shared but concepts conflicting.

H3e: No properties are shared.

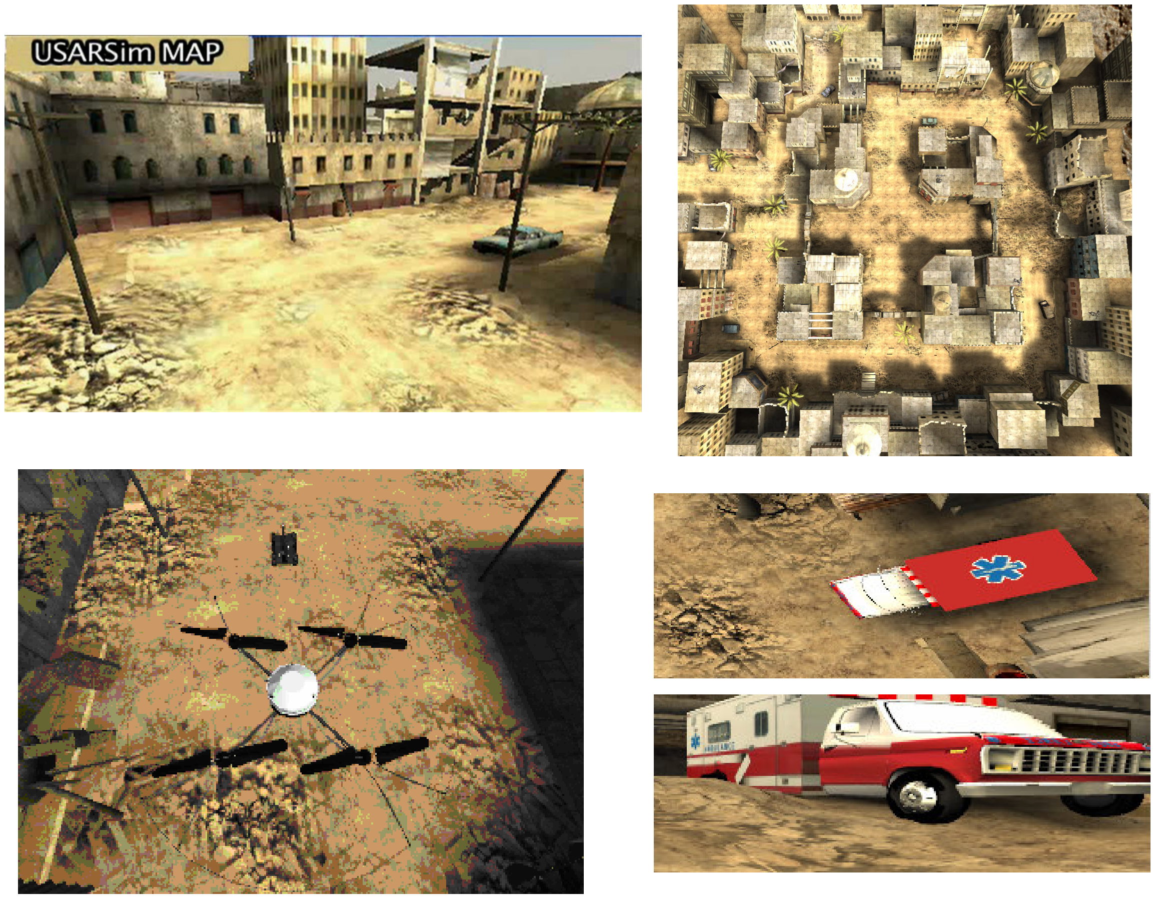


Figure 29 – Upper Left: Image showing the USARSim simulation environment. Upper Right: View of the entire village from above. Lower Left: Ground Talon robot and aerial quadrotor robot used in the experiments. Lower Right: An object (ambulance) from the perspective of the ground robot (upper right) and aerial robot and (lower right).

In this dissertation, we will show how two robots can deal with different types of heterogeneity that they have through interaction in the environment, and map properties and concepts that they do share. We then show how the resulting mappings can be used in tasks involving knowledge exchange and adaptation. Note that heterogeneity types H1b, H1c, H2c, and H3c will not be dealt with in this dissertation due to scope. It is anticipated that the more common situation will consist of properties that overlap fully or partially, although exploration of these additional types of heterogeneity may be an interesting scientific endeavor.

Figure 30 – All eight objects used in the USARSim simulation experiments.

3.7. Experimental Platforms

The principles of this dissertation will be tested with simulation and real-robot experiments. Three different configurations of robots and sets of objects were used, one of which was in simulation with the other two consisting of real robots.

3.7.1. Simulated Platforms

The first configuration is a ground/aerial robot combination in the USARSim 3D simulation environment (Carpin et. Al, 2005). The robots and environment are shown in Figure 29. USARSim is a realistic 3D simulation environment developed by NIST for search and rescue competitions, and is based on the Unreal Tournament 2004 game engine. As such, in incorporates realistic shading and lighting, which adds variability to the objects in the environment. It also includes models of most modern robot and sensor platforms. In this case, we use two robots, a ground tracked robot modeled after the Talon robot and an aerial quadrotor robot. In this situation, there is an inherent heterogeneity in perspectives, as the two robots have very different points of view of the same objects.

The environment consisted of an outdoor village containing a large number of buildings and objects. Eight objects, including various cars, vans, and large objects were used to test the learning and transfer processes (Figure 30). Color and texture properties,

Figure 31 – Pioneer 2DX robots used in some of the experiments (left) and images of the same scene from each robot (middle and right). The middle image is from the robot with the web camera, while the image on the right is from the robot with the camcorder.

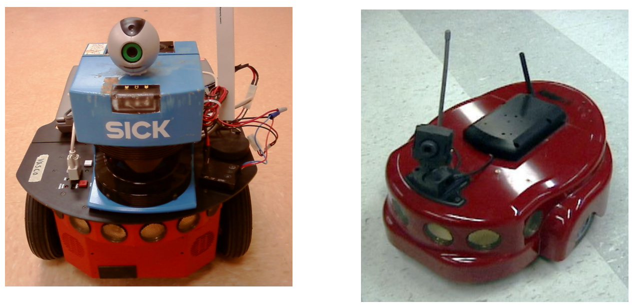


Figure 32 – Pioneer 2DX robots (left) and Amigobot (right) robots used in some of the experiments.

obtained from images, were used to represent objects. Color was represented as median values of the object in RGB or HSV color spaces. These values were then generalized in the form of our Gaussian Mixture property representation. Texture was represented using the mean and standard deviation of the output of a single empirically-derived Gabor filter over the entire object. The simulated robots were used to perform various tasks in the outdoor environment depicted in Figure 29. This environment allowed us to test applications of the framework laid out in this dissertation to a joint reconnaissance scenario, where the two robots explored the outdoor environment and were tasked to find and follow various moving objects.

Figure 33 – Twelve objects used in some of the real-robot experiments.

3.7.2. Real-Robot Platforms (Configuration 1)

The first real-robot configuration consisted of two Pioneer 2DX robots, seen in Figure 31. The first robot (left) had odometry, sonar sensors, and a Quickcam Express web camera. The second robot (right) had odometry, sonar sensors, and SICK ladar sensor, and a DCR-HC40 Sony Handycam camcorder. In this case, only the cameras were used; range-sensing is used in the next configuration to obtain object characteristics such as size and shape. Again, the same color and texture properties were used to represent objects.

3.7.3. Real-Robot Platforms (Configuration 2)

In the final configuration, consisting of real robots, a Mobile Robots Amigobot with a wireless camera and a Pioneer 2DX robot with a Quickcam Express web camera were used. These robots are shown in Figure 32. These robots are the same that were used during the experiments earlier in this chapter, except in that case only twelve of the thirty-four objects were used. These twelve objects, representing a subset of all of the objects used, are shown in Figure 33. As can be seen, the objects varied in color, texture, size, and shape.

The Pioneer with the web camera used 320x240 resolution images while the other used 640x480 resolution, adding another source of heterogeneity. Unlike the first robot, the Pioneer robot with the web camera had a SICK laser range finder as well, data from which was processed to extract object shape (curvature) and size properties. Color and texture properties were derived similar to the other configurations.

Shape, obtained from range finders, consisted of a curvature metric. Object sizes were obtained by simply calculating the 3D points in which they lied and measuring the three dimensional distance between the first point on the object and the last. The camera and laser sensors were calibrated so that points from the laser could be projected onto the image.

In some cases, these features (and therefore properties) were missing if, for example, the object was not in full view and the laser readings therefore did not cover the entire object. In these cases, the features and property memberships were considered missing; as mentioned, the ability to handle missing features is an important capability to enable transfer learning of classifiers.

3.7.4. Processing

Most of the processing was performed in MATLAB offline, although this is not due to processing requirements. Images were gathered from the robots and separated into testing and training sets. These images were segmented using the automatic segmentation

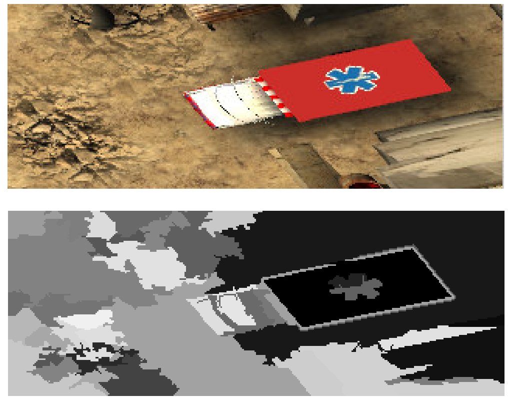


Figure 34 – Example segmentation of an image. The top image shows the original image while the bottom image shows the resulting segmentation. Different shades of gray were used to depict different segments.

algorithm (Felzenszwalb and Huttenlocher, 2004). Figure 34 shows an example. Where supervised learning was used, segments were chosen in MATLAB by the user for each corresponding object. Calibration was done using checkerboard patterns and the MATLAB Calibration Toolbox. The 3D point cloud resulting from the laser readings were then calculated, and the camera calibration was then used to project the laser points onto the image. Points that were inside the segments of the object were then used to determine shape and size as described previously.

3.8. Experimental Results: Property and Concept Learning

We now detail our results with respect to the learning of properties and concepts. This first experiment serves to show that properties can be learned from data, conceptual spaces can be used to combine these properties and learn object models, that performance increases with additional training samples indicating successful learning, and that the object models can then be used to classify objects. Table 7 summarizes the experiment.

Table 7 - Experimental summary for the experiment demonstrating property and concept learning using conceptual spaces.

Experiment 2: General Experiment Summary Property and Concept Learning	
Purpose	To show that conceptual spaces is a viable representation that can be learned using real data.
Experiment Type	Simulation, Real-Robot (Configuration 2)
Hypothesis	We hypothesize that classification rates will be significantly better than chance, and that as the number of training instances increases performance will improve (until reaching a plateau)
Procedure	1. Gather and label sensory data 2. Train properties using labeled instances 3. Train concepts using labeled instances 4. Test learned concepts on testing data 5. Gather classification accuracy results
Performance Metrics	Areas under curve for: ROC curves, recall learning curves, precision learning curves.

3.8.1. Hypothesis

The hypothesis of this experiment is that the property abstractions and conceptual spaces representation can be used to classify objects. Specifically, we hypothesize that classification rates will be significantly better than chance, and that as the number of training instances increases, performance will improve until reaching a plateau. While basic in nature, these experiments serve to show that conceptual spaces is a viable representation that can be used for real-world data. Other hypotheses regarding the difficulty of naïve knowledge transfer on heterogeneous robots (Section 3.9) and the importance of the property abstraction for learning (Section 3.10) will follow.

3.8.2. Performance Metrics

We measure performance using receiver operator curves (ROC), recall rates, and precision rates. The ROC plots show the true positive ratio against the false positive ratio. The true positive ratio, or sensitivity, of a binary classifier is the number of true positives divided by the total number of positives in the test set. The false positive ratio is the number of false positives divided by the total number of negatives in the test set. The recall rate measures the number of true positives divided by the number of positives in the test set, while precision measures the number of true positives divided by the number of test set instances that were classified to be positive. These numbers are proportions and can, if desired, be represented as percentages. For recall and precision, learning curves are plotted showing the average performance over all concepts as the robot is learning and the number of training instances increases. For all of these metrics (ROC, recall learning curves, and precision learning curves), the area under the curves can be used to quantitatively assess the robots' performance. The latter two (area under the recall and precision learning curves), in particular, assess the robots' concept classification performance throughout its entire learning process.

Table 8 - Properties and objects used to represent concepts.

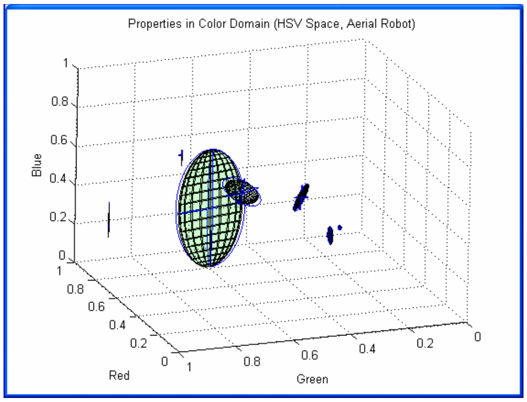
Property # Number		Objects Included	Semantic Label
Ground	Aerial		
p_3	p_1	Mailbox Trash Police Car	Blue
p_1	p_2	Van	Brown
p_2	p_3	Ambulance Barricade Cone Race Car	Red
p_4	p_4	Mailbox Barricade Police Car	Smooth
	p_5	Van (3 types)	Textured

3.8.3. Simulation

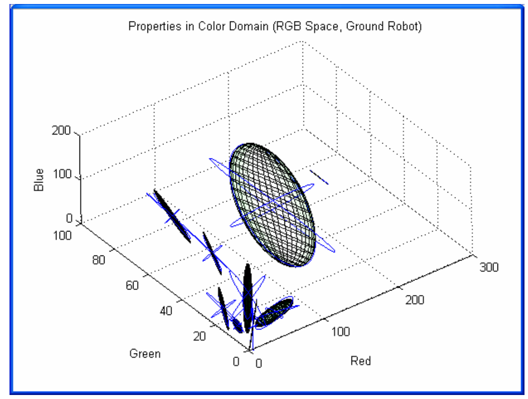
3.8.3.1. Procedure

We will first describe the simulation experiments for learning of properties and concepts. These results show that our representation can be used to learn about and classify objects in the world. In order to gather data, the simulated robots were teleoperated to view the various objects. The aerial robot flew approximately eight meters from the ground. A portion of the environment and the two robots can be seen in Eight objects were used for testing property and concept learning as well as transfer, all of which can be seen from the perspective of the ground robot in Figure 30. A large number of images containing each of the objects in many different perspectives were gathered. For the van object, three different instances of the van in different lighting conditions were used to train the properties. The objects used were realistic, challenging, and were found under varied lighting. Out of these, 70 randomly chosen images were used for training, and 30 (different) randomly chosen images were chosen for testing. All images were automatically segmented using a graph-based image segmentation algorithm (Felzenszwalb and Huttenlocher, 2004). An example segmentation can be seen in Figure 34.

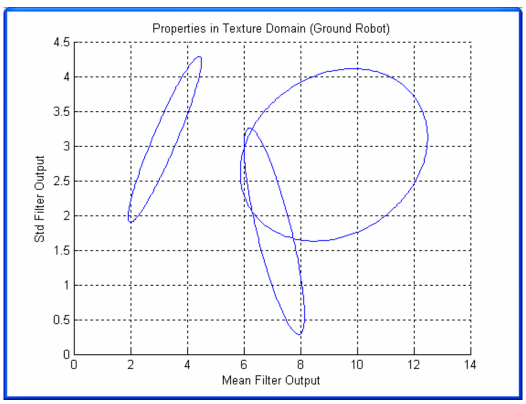
In order to train the properties, each object was categorized as belonging into one of three color properties (see Table 8). Both robots were trained with instances containing these properties, but the ground robot used an RGB color space while the aerial robot used an HSV space. The ordering of the properties was randomized for each robot. For texture, both robots used the same two dimensional space consisting of the mean and standard deviation output of a Gabor Filter. However, the ground robot only had one texture category corresponding to smooth objects such as the barricade or mailbox, while the aerial robot had an additional property corresponding to the texture pattern of the van (object 7 in Figure 30). Hence, the ground robot had four total properties while the aerial robot had five (heterogeneity type H2d). Note that not all objects were used in training of all of the properties (e.g. some objects were not used to train texture properties). While in simulation there is no heterogeneity in the cameras themselves, in this case



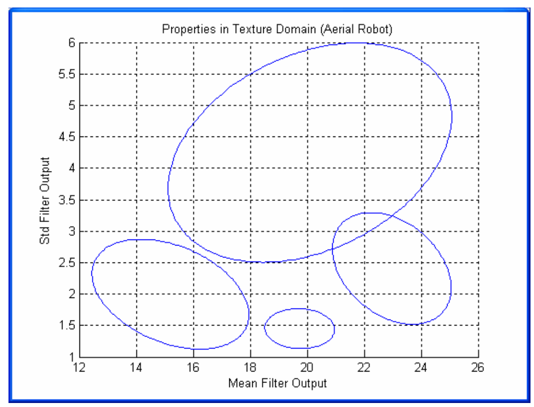
(a) Color properties for the aerial robot (HSV color space).



(b) Color properties for the ground robot (RGB color space).



(c) Texture properties for the aerial robot. The space consists of the mean and standard deviation output of a Gabor filter.



(d) Texture properties for the ground robot. The space consists of the mean and standard deviation output of a Gabor filter.

Figure 35 – Color and texture properties, represented as a Gaussian Mixture Model, after training with multiple objects. This figure is meant to demonstrate what was learned in the simulation experiments with respect to properties.

heterogeneity originates from utilizing different metric spaces, one robot using an additional property that the other did not have, and large differences in perspectives which led to different portions of the objects being used during training.

For each property, 70 images of each object having that property were used to train the Gaussian Mixture Model (GMM). During training, a segment corresponding to the target property as well as the type of property being trained (e.g. color or texture) were

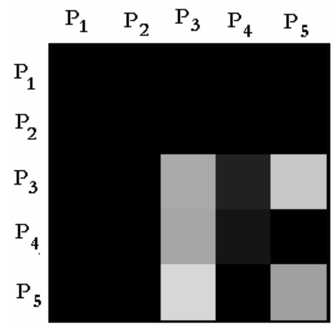


Figure 36 –Concept matrix for the race car object (aerial robot). Lighter values indicates higher correlations between properties. High values can be seen for property 3 (red) and property 5 (corresponding to textured as opposed to smooth).

hand-labeled. The target segment was then processed (e.g. median RGB, median HSV, or texture output values calculated for that segment), and the resulting set of data points were used for training the GMM. The algorithm for property training that was used is described in Section 3.5 and specifically the algorithm in Table 5. We used up to three clusters per property in this case, with the actual number determined by a minimum description length criteria applied to the Expectation Maximization algorithm (Bilmes, 1998). Figure 35 shows the resulting properties for both color and texture properties.

After the property representations were learned, there was a second training period when the concepts (i.e. objects) themselves were learned. Concept learning was performed as described in Section 3.5 and specifically using the algorithm in Table 6. In this case, the incremental version of concept learning was not used since there was much less noise in simulation. Instead, median property membership values were used. Again, 70 images were used per concept along with a target segment that contained the target concept. As described previously, each concept was represented via a matrix containing correlations between each pair of properties. As an example, Figure 36 shows a gray-scale depiction of the concept matrix for the race car, where brighter values correspond to higher values (values range from 0 to 1, inclusive). As can be seen, high values were seen for property 3 (red) and property 5 (corresponding to textured as opposed to smooth). Correlations between properties 3 and 5 were seen as well, since whenever the race car had a high red property it also had a high value for the textured property (i.e. there was not much variety in the appearance of the race car).

After the concepts were learned (i.e., all of the training instances were used), we tested the accuracy of categorization of six hundred images, some of which contained the eight objects but many of which did not contain any of the learned concepts. In total, 600 images were used for testing, only 30 of which contained any one trained object. Since each image contained 34 segments on average in the segmentation, this is a challenging categorization task as there can be many small segments that do not contain many pixels and spuriously led to false positives.

In this case, the process was fully automated: each image was segmented, the property membership values of each segment were calculated (as described in Section 3.5), and finally the concept membership values were calculated using the algorithm in 3.4, namely Table 4. This algorithm was run for all concepts, resulting in a list of concept memberships (one for each learned concept) per instance.

Detection and categorization accuracy was then determined using standard receiver operating characteristic (ROC) curves, which plots the true positive ratio against the false positive ratio. Thresholds on the concept membership value were varied from 0 to 1 in increments of 0.005. Each threshold value was used to decide whether a segment corresponded to a given concept or not. These classifications are then compared to ground truth (which is known as the data is labeled), resulting in statistics such as false positive and negative rates. In the ROC curve, as the threshold is loosened, more true

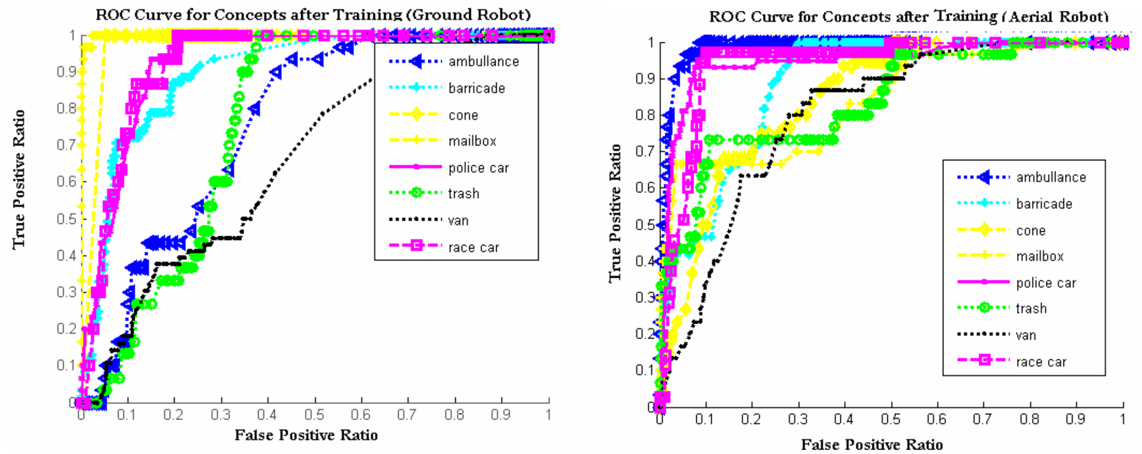


Figure 37 – ROC curve for eight concepts on aerial and ground robot. A classifier performing at chance would yield a diagonal line between (0,0) and (1,1). These results show successful classification better than chance. The mean area under the ROC curve for all of the objects was 0.82 (left) and 0.89 (right), significantly better than chance performance of 0.5.

positives are obtained but potentially resulting in higher false positives. Each false positive and true positive value was then plotted. The best possible classifiers would lie at the upper left corner, corresponding to only true positives and no false positives. A classifier performing at chance would lie on the diagonal line going from (0,0) to (1,1). One measure of total accuracy that can be used is the area under the ROC curve, where a value of one is perfect. A classifier performing at chance would result in an area of 0.5.

<u>3.8.3.2.</u> <u>Results</u>

We now describe the classification results, in the form of ROC curves. Figure 37 shows the curves for the two robots; the mean area under the ROC curve for all of the objects was 0.82 for the ground robot and 0.89 for the aerial robot, representing good categorization results given the challenges of using automatic segmentation and a large number of test images that did not contain any of the learned concepts. On average, in the simulation results, the aerial robot categorized objects better, as HSV is a more effective color space and objects viewed from above vary much less than from below where occlusion and perspective differences can be a problem. These results confirm our hypothesis that conceptual spaces can be used to classify objects at rates significantly better than random: 0.82 and 0.89 ROC areas for the ground and aerial robots, respectively, compared to 0.5 for random chance.

3.8.4. Real-Robot (Configuration 2)

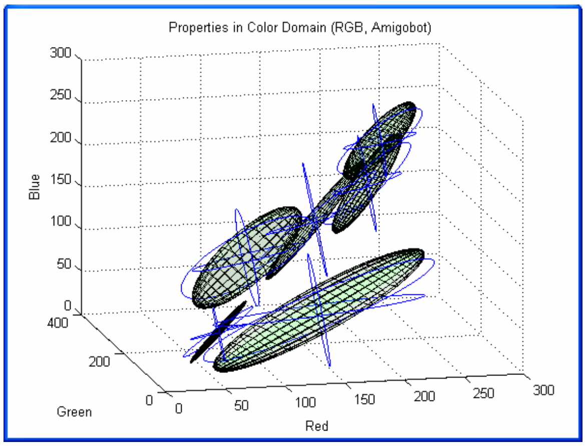
3.8.4.1. Procedure

We now describe a similar experiment as in Section 3.8.3, but in this case using real robots (configuration 2). Just as before, in order to train color, texture, shape, and size properties, the robots were driven around a laboratory environment, resulting in a large amount of stored sensor data. Thirty-four realistic objects were used, twelve of which were shown in Figure 33. Examples of objects included whiteboards, wood crates, some robots (e.g. a different Amigobot as well as an iRobot Create), trash cans, and so on.

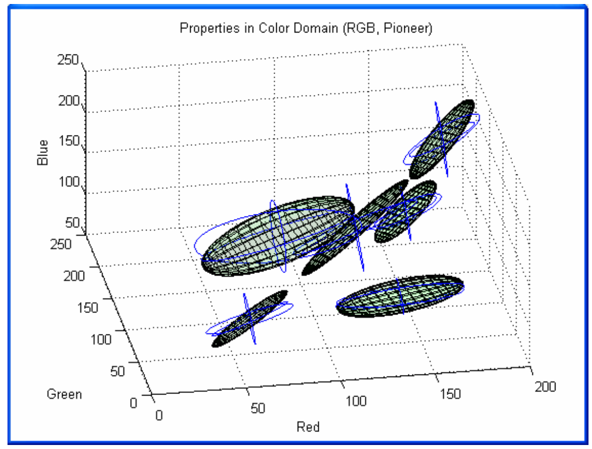
Table 9 -Properties and example objects used to train them.

Property	Objects Incl.
White	Whiteboard, Create Robot
Tan	Crates
Black	Chair1, Case, Trash
Gray	Fence, Cabinet
Blue	Chair2, Recycling Bin
Texture1	Chair2, Black Case
Texture2	Gray Cabinet, Fence
Texture3	Couch
Texture4	Crates
Round	Recycling Bin, Trash
Flat	Whiteboard, TV Box
Small	Bucket, Cooler
Large	Whiteboard, Crate

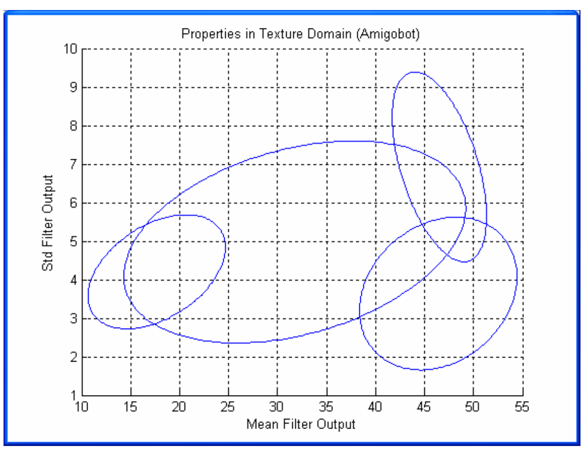
One hundred images of each object were chosen, seventy of which were randomly chosen for training and thirty for testing (note some object categories had somewhat fewer testing instances). Anywhere from one to six objects from the environment per category were chosen for training of properties, some of which are shown in Table 9. For each property, all that is given is the domain to be trained, a set of data instances, and segments chosen from the automatic segmentation of the target object. The color space used for the properties included RGB and HSV. For texture, an empirically-chosen Gabor filter was used, with



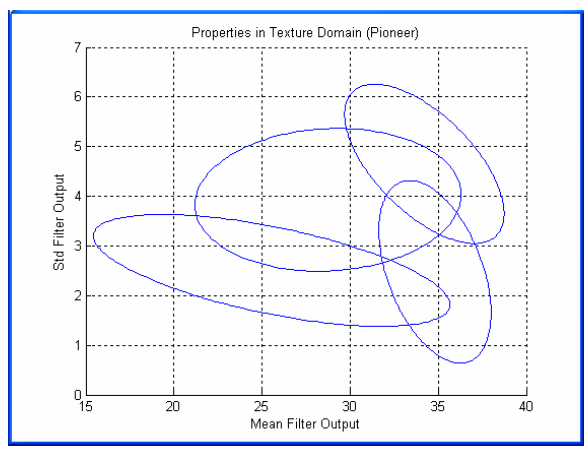
(a) Color properties for the Amigobot (RGB color space).



(b) Color properties for the Pioneer (RGB color space.



(c) Texture properties for the Amigobot. The consists of the mean and standard deviation output of a Gabor filter.



(d) Texture properties for the Pioneer. space

Figure 38 – Color and texture properties, represented as a Gaussian Mixture Model, after training with multiple objects. This figure is meant to demonstrate what was learned in the real-robot experiments (robot configuration 2) with respect to properties.

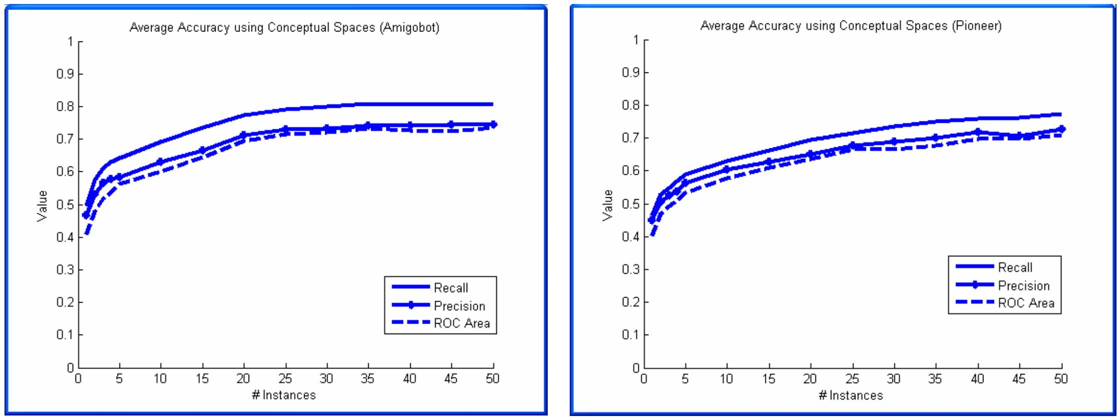


Figure 39 – Results demonstrating successful concept learning for the Amigobot (left) and Pioneer (right). Classification results are greater than chance (0.5) for all three metrics, and performance increases as more training instances become available indicating successful learning.

the mean and standard deviation of its output comprising the space. Shape, obtained from range finders, consisted of a curvature metric. Object sizes were obtained by simply calculating the 3D points lying on the objects and measuring the three dimensional distance between the first point on the object and the last. The camera and laser sensors were calibrated so that points from the laser could be projected onto the image.

Properties were trained using this data using the EM algorithm described in Section 3.5, specifically the algorithm in Table 5. Figure 38 shows the resulting learned properties. After the properties were trained, the concepts were trained using labeled data, again described in Section 3.5, specifically the algorithm in Table 6. Again, the concept membership algorithm was run for all concepts, resulting in a list of concept memberships (one for each learned concept) per instance. Recall and precision rates were calculated by taking the maximal concept membership for each instance and comparing it to the known ground truth concept label. The recall and precision results were calculated for different number of training instances in order to plot learning curves. Specifically, rates for the first five training instances were measured, after which results for every subsequent five instance was measured (up to fifty, the total number of training instances).

3.8.4.2. Results

Figure 39 shows learning curves for concept learning using conceptual spaces. Each instance was used to update the concept matrix, and after instances one through five, and every five instances thereafter, accuracy was measured on a test set using manually-specified ground truth. As can be seen, both robots achieve better than chance performance after seeing all training instances (0.81 recall rate, 0.74 precision rate, and 0.73 ROC area for the Amigobot; 0.77 recall rate, 0.73 precision rate, and 0.71 ROC area for the Pioneer), with recall being higher than precision. A classifier performing at chance would have a recall and precision rate of 0.5. Furthermore, as the number of training instances increases the recall and precision rates improved before reaching a plateau, demonstrating successful learning. In Chapter 5, these same curves will be shown again but augmented with the learning curves obtained after knowledge transfer. These results confirm our hypothesis that learning is occurring, as the average accuracy (as measured by recall and precision) increased as the number of training instances increased.

3.8.4.3. Discussion

We have demonstrated in this subsection that the use of conceptual spaces is a viable method for representing real-world concepts that are sensed by robots through noisy sensors. Classification results are significantly better than chance, and performance increases through further training until a plateau is reached, demonstrating learning. We have shown this through both simulation and real-robot experiments. The resulting properties (which were plotted throughout) and concepts learned in these experiments will be used throughout the dissertation to validate further hypotheses regarding the nature of learning and knowledge transfer. Table 10 shows a summary of the experiment.

Table 10 - Experimental summary and conclusions for the experiment demonstrating property and concept learning using conceptual spaces.

Experiment 2: General Experiment Conclusions **Property and Concept Learning**	
Hypothesis	We hypothesize that classification rates will be significantly better than chance, and that as the number of training instances increases performance will improve (until reaching a plateau)
Conclusions	Hypothesis is confirmed. Classification results after training are better than chance (simulation and real-robot experiments) and learning curves demonstrated improved performance as the number of training instances increased (real-robot experiment).

3.9. Experimental Results: Heterogeneity and Direct Property Transfer

3.9.1. Hypothesis

We now provide further evidence for our hypothesis that heterogeneity can pose difficulty for knowledge transfer when performed without *a priori* modeling of differences (Kira, 2009a). The hypothesis is that heterogeneity poses a problem during uninformed knowledge transfer. In other words, attempting knowledge transfer without understanding robot differences may result in unsuccessful transfer. We have already demonstrated this using a best case scenario (Section 3.1), but will further show this using our representations as well, in this case showing that robots could not successfully classify property categories using properties obtained from another heterogeneous robot. Specifically, we hypothesize that transferring properties directly between heterogeneous

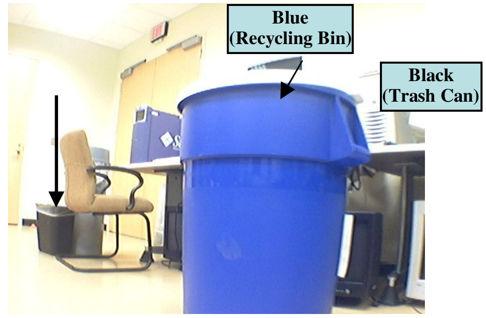


Figure 40 – Example objects used for testing.

robots will significantly reduce the classification rate of property categories. In other words, before concepts are even represented, heterogeneity can present a problem during the transfer of just the underlying properties.

3.9.2. Procedure

For these experiments, we used two real robots (configuration 1). In order to show that direct transfer or comparison of properties across heterogeneous robots can result in a degradation of performance, we directly transferred the learned GMM models from robot 1 to robot 2, and vice-versa, for the same RGB color space. We then tested the resulting categorization success in the same manner as before. In other words, robot 1 used robot 2's learned representation on its own data in order to categorize the testing set.

The process of training properties was similar to that to that of the simulated experiments. Specifically, we used the methods described in Section 3.5, particularly the algorithm in Table 5. We used up to three cluster per property in this case, with the actual number determined by a minimum description length criteria applied to the Expectation Maximization algorithm (Bilmes, 1998). In order to train color and texture properties, the robots were driven around a laboratory environment for 2-3 runs, resulting in a large amount of stored sensor data. Six to eight objects from the environment per color category were chosen for training. For texture, a single empirically-chosen Gabor filter was used, with the mean and standard deviation of its output comprising the space. Examples of objects include a blue recycling bin, a smaller black trash can, and a blue Sun computer, all of which can be seen in the images in Figure 40. In order to avoid bias, property numbers were randomly assigned to the actual color or texture that was used during training. It is important to note that symbolic labels were not given to the robots, they are only added by the author for clarification of the figures. For each property, all that is given is the domain to be trained (i.e., whether the property is in the color or texture domain), a set of data instances, and segmentation of the target object. In other words, the robots could not simply compare labels to determine which of their properties mapped. Table 11 shows the assignments that were given to both robots for the color and texture properties. Knowledge of this mapping is not used by the algorithms, and is what must be learned by the robots given instances from a shared context in Chapter 4.

Table 11 - Table of arbitrary symbols assigned to color categories. The same symbol (e.g. "Black") was arbitrarily assigned as a differently numbered property for the two robots to avoid bias.

	Brown Objects	Black Objects	Blue Objects	Gray Objects	White Objects
Symbol: Robot A	p_1^A	p_2^A	p_3^A	p_4^A	p_5^A
Symbol: Robot B	p_1^B	p_3^B	p_5^B	p_2^B	p_4^B

Out of all of the images recorded, 45 were chosen per category containing an approximately equal number of instances from each object in the category, resulting in a total of 225 images. 150 of these were randomly chosen for training, while the rest were used in testing the categorization and building the confusion matrices. The same images were used for training of texture properties. The objects were segmented manually in the

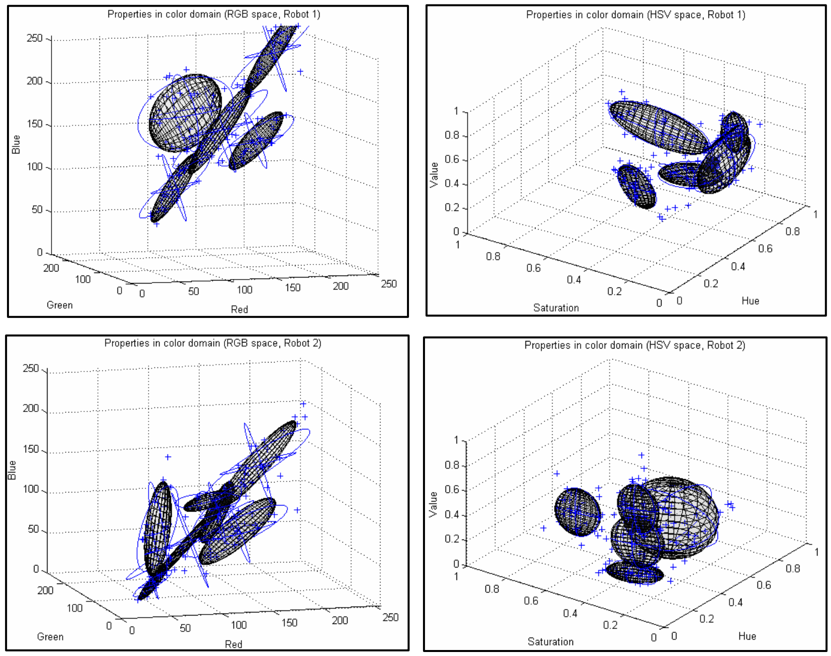


Figure 41 – Color properties, represented as a Gaussian Mixture Model, after training with multiple objects with five colors. Results are shown for two color spaces (RGB and HSV) and the two heterogeneous robots. The resulting models are significantly different for each robot, arising due to heterogeneity in training and sensing.

image, differing from the automatic segmentation performed in experiments using the other platforms.

3.9.3. Results

Figure 41 shows the resulting property regions for both robots in two color spaces (RGB and HSV). As can be seen, despite being trained on the same objects, the representations are quite different. After training the properties using supervised learning, the accuracy of categorizing the color of different views and instances of the objects was tested. Table 12 shows the resulting accuracy results for both robots, averaged over five runs (standard deviations are shown). They both achieved better than chance accuracies (80.8% and 79.7% for robot 1 and 2, respectively) given the existence of object brightness changes due to changes in perspective. Interestingly, the results for robot 1 (that had an inexpensive web camera) performed similarly to the second robot that had a more expensive camcorder. Overall, the camcorder resulted in colors that were duller and less bright, as can be seen from the results in Figure 41.

Table 12 - Color categorization accuracy with and without transfer. Categorization was significantly better than chance when the robot used its own representation, but at chance levels when the robot used transferred properties received from the other robot.

Robot	Own Representation		Transferred Representation	
	# (/ 75)	Percent	# (/ 75)	Percent
Robot 1	60.6	80.8 ± 5.0	14.4	19.2 ± 0.7
Robot 2	59.8	79.7 ± 2.9	16.4	21.9 ± 2.6

Table 13 - Experimental summary and conclusions for the experiment demonstrating the failure of direct property transfer due to heterogeneity.

Experiment 3: General Experiment Summary & Conclusions **Heterogeneity and Direct Property Transfer**	
Purpose	To determine whether heterogeneity can pose difficulty for knowledge transfer when performed without *a priori* modeling of differences.
Experiment Type	Real robot (configuration 1)
Hypothesis	Transferring properties between heterogeneous robots will significantly reduce the classification rate of property categories by the receiving robot.
Procedure	1. Train properties for each robot 2. Determine classification rate for property categories. 3. Transfer properties between robots 4. Determine new classification rate for property categories when using received properties. 5. Compare results from (2) and (4).
Independent Variable	Source of property models (self-learned or received)
Dependent Variable	Performance during classification of property categories.
Conclusions	Hypothesis is confirmed. Using properties received from another robot was significantly worse for classification of property categories than properties learned by the robot itself.

The right side of Table 12 shows classification results when the properties were transferred between the robots and used by the receiving robot to similarity classify the object property. As can be seen from the right side of Table 12, the results were dramatically worse (19.2% and 21.9% for robot 1 and 2, respectively) and close to random guessing (which would achieve a 20% accuracy). The difference is significant for both comparisons ($p<0.0001$ for both). Hence, even with training sets consisting of the same objects and when using the same color space the properties of the two robots were incompatible due to sensor heterogeneity. We have now shown multiple situations where naïve transfer of knowledge can fail: The experiments in Section 3.1 and the experiments in this subsection. Table 13 summarizes the experiment in this subsection.

3.10. The Importance of Property Abstractions for Learning

The role of properties as an intermediate level of abstraction is important to the conceptual spaces representation. They form the bridge between raw sensory features to higher level concepts. However, one can ask whether the abstraction of data in the form of properties is useful. In this dissertation, we hypothesized two useful roles for property abstractions: 1) We hypothesized that property abstractions will aid learning; i.e. that learning will be easier and 2) We hypothesized that properties will serve as a buffer for heterogeneity between robots, such that similar properties learned by the robots will have similar memberships to the same concept, despite differences in the actual data and domains. In this section, we will provide evidence for the first hypothesis (Kira, 2010). We will then do the same for the second hypothesis in Chapter 5 which deals with knowledge transfer (specifically, Section 5.6).

3.10.1. Hypothesis

For this experiment, the hypothesis is that our method of sensory abstraction in the form of properties will aid the learning of object representations when compared to using the raw sensory data instead.

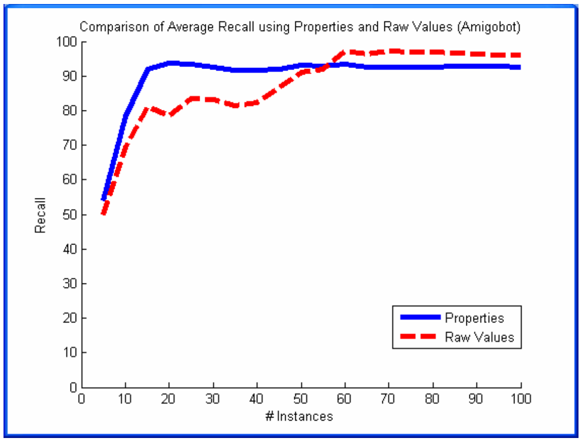
3.10.2. Procedure

In order to test the hypothesis that these property abstractions improve learning, we performed experiments using a support vector machine (SVM) classifier to learn and classify objects (specifically, we used the svm^{light} software package (Joachims, 1999)). In this case, we use a standard machine learning algorithm since it allows the robot to learn with raw sensory data as a comparison. We hypothesize that learning with such data would be more difficult to do without our property abstraction when using conceptual spaces. Furthermore, it again provides a best case scenario, since we use a discriminative learner that finds optimal separation from other objects. Discriminative learning (such as SVMs) often performs better than a generative learner (which conceptual spaces falls under), although it requires both negative and positive examples. In addition to confirming our hypothesis, by using support vector machines (a popular technique in the machine learning community), these experiments indicate that many elements of the work in this dissertation are not necessarily tied to the representation of objects used, bolstering the generality of our work. Specifically, the property abstraction methods can be combined with other classification methods (besides conceptual spaces), and hence the methods for mapping such properties between robots (described in Chapters 3 and 5) can also be applied to other object representations.

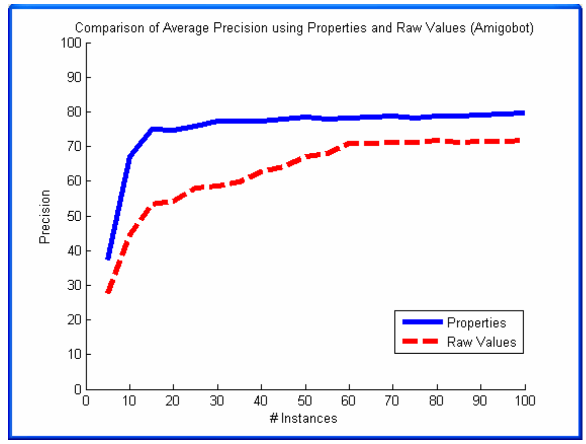
Two conditions were used in this experiment. In the first condition, the property memberships for the previously learned properties were used as attributes. In the second condition, raw sensory data itself (e.g. RGB values or curvature metric) were used as attributes for training. In later chapters, we also used a third condition where raw sensory data was used, but the robots used different representations for color, namely one robot used an RGB color space while the other used an HSV color space. In order to gauge classification rates, both recall and precision are plotted as the number of training instances increases. These are standard classification metrics, where recall measures the number of true positives divided by the number of positives in the test set, while precision measures the number of true positives divided by the number of test set instances that were classified to be positive.

3.10.3. Results & Discussion

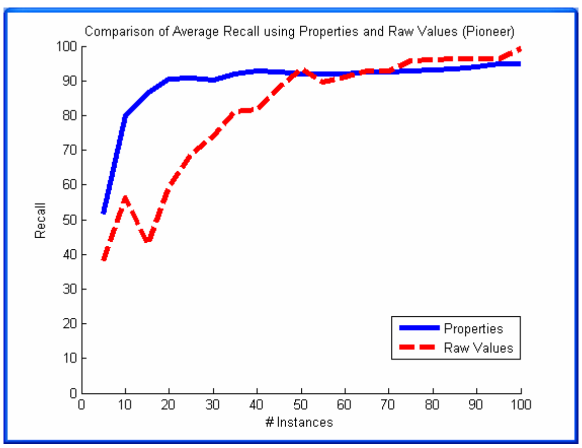
Figure 42 shows a comparison between the first two conditions for both robots and both recall and precision. As can be seen, our hypothesis that the abstraction of properties results in higher learning curves is confirmed, showing that it is more difficult to learn with raw sensory data. Final recall rates after all training instances are a little higher when using raw sensory data (96.0% when using raw values compared to 92.7% when using properties, or a 3.4% decrease) but this comes at the expense of lower precision (71.8% when using raw values compared to 79.6% when using properties, or 10.9% increase). Furthermore, the learning curve is higher when using properties most



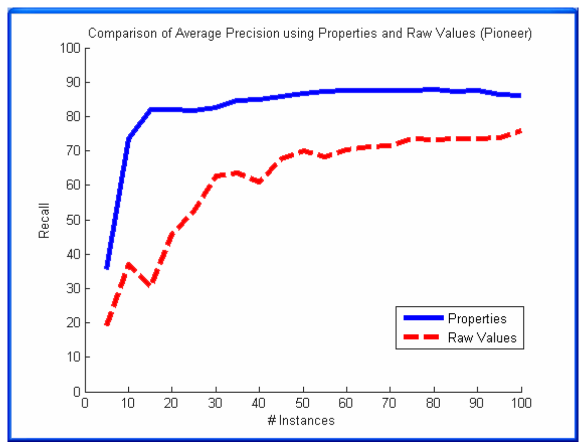
(a) Recall learning curve for the Amigobot. The areas under the curve were 0.86 when using properties and 0.84 when using raw values.



(b) Precision learning curve for the Amigobot. The areas under the curve were 0.72 when using properties and 0.61when using raw values.



(c) Recall learning curve for the Pioneer. The areas under the curve were 0.86 when using properties and 0.78 when using raw values.



(d) Precision learning curve for the Pioneer. The areas under the curve were 0.80 when using properties and 0.59 when using raw values.

Figure 42 – Results demonstrating the advantage of using abstracted properties as opposed to raw sensory data when learning. These learning curves, showing performance (y-axis) as the robot continues to learn and the number of training instances increases (x-axis). Each subfigure shows higher curves (as measured by areas under the learning curves) to demonstrate that learning with property abstractions is easier. The figures on the left show precision, while the figures on the right show recall. The figures on the top show results for the Amigobot robot, while the figures on the bottom show results for the Pioneer 2DX robot.

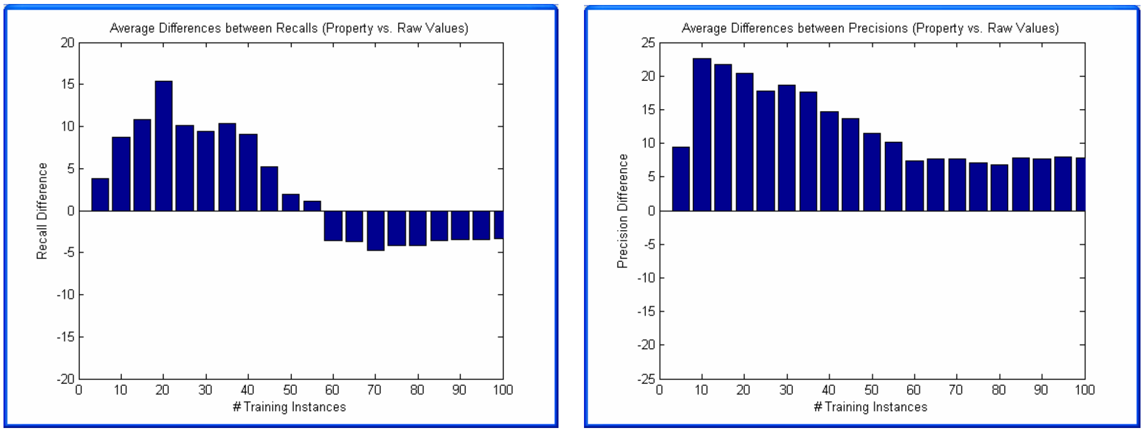


Figure 43 – Results demonstrating the advantage of using abstracted properties as opposed to raw sensory data when learning. The figure on the left shows precision, while the figure on the right shows recall. Positive values indicate larger recall and precision rates when using property abstractions, while negative values indicate larger rates when using raw values. The graph is dominated by positive values, indicating that learning is easier using property abstractions. This is validated quantitatively by measuring the areas under the learning curve in Figure 44.

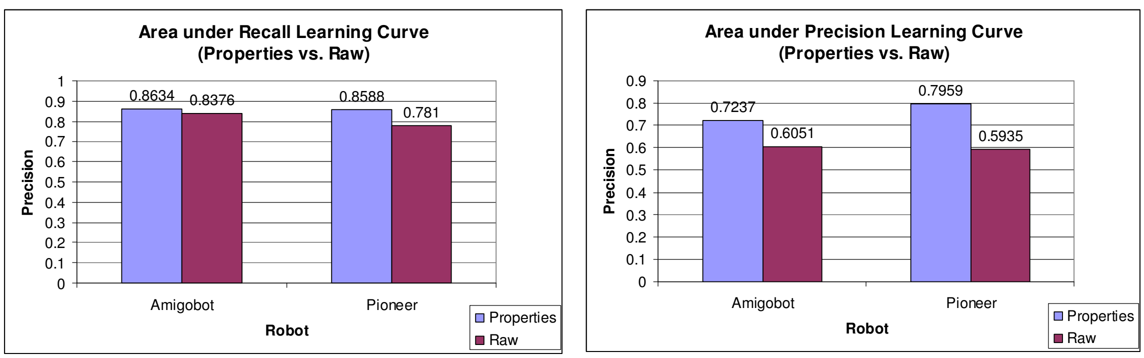


Figure 44 – This graph shows the areas under the learning curves for the two robots in the two experimental conditions (property abstractions vs. raw values). Higher values are achieved when using property abstractions, indicating that learning is easier when using them when compared to raw sensory data.

of the time (and especially in the early stages of learning), showing that it is more difficult to learn with raw values. This can be seen more starkly for the Amigobot robot in Figure 43, where we plot a histogram of the differences between using properties and using raw values. These values indicate the raw increase in rates when using properties over when using raw values. We measured the overall gain quantitatively by measuring the area under the learning curve. This metric measures the overall classification accuracy over the entire training process of the robot and as it obtains more training instances. Figure 44 summarizes the results for both recall and precision, for both robots. Areas under the learning curve were smaller for both recall (e.g. 0.86 versus 0.84

Table 14 - Experimental summary and conclusions for the experiment demonstrating the importance of property abstraction for learning.

Experiment 4: General Experiment Summary **The Importance of Property Abstractions for Learning**	
Purpose	To determine whether the abstraction of raw sensory data into property abstractions improves learning.
Experiment Type	Real-robot (configuration 2)
Hypothesis	We hypothesize that property abstractions do indeed aid learning. Specifically, that the learning curves when using property memberships to model and classify objects will be higher than when using raw sensory data.
Procedure	1. Train properties using labeled data. 2. Train two classifiers for concepts using labeled data A. The input attributes to the classifier is raw sensory data. B. The input attributes to the classifier is property memberships.
Independent Variable	Input attributes to the classifier (raw sensory data vs. property memberships)
Dependent Variable	Accuracy of concept classification as measured by recall, precision, and areas under the resulting learning curves as the number of training instances increases.
Conclusion	Hypothesis is confirmed. The learning curves for both robots are higher when using property abstractions compared to using raw sensory data.

for the Amigobot when using properties and raw values, respectively) and precision (e.g. a 0.72 recall rate versus 0.61 for the Amigobot when using properties and raw values, respectively, and 0.86 precision rate versus 0.78 for the Amigobot when using properties and raw values, respectively). For the Pioneer robot, the recall rate area was 0.72 when using properties and 0.60 when using raw sensory data, while the precision rate area was 0.80 when using properties and 0.59 when using raw sensory data.

Overall, the data presented in this experiment has shown that it is easier to learn with properties than raw sensory data, as determined by areas under the learning curves. In the next chapter, dealing with concept transfer, we will further show that property abstractions also aid transfer. That is, transfer between heterogeneous robots can sometimes fail when using raw sensory data as features, but not when using property abstractions. Table 14 summarizes the experiment in this subsection.

3.11. Summary

This chapter laid the foundation for the dissertation. We began with an experiment utilizing three robots and thirty-four real world objects. We empirically demonstrated that perceptual heterogeneity does indeed prevent naïve knowledge transfer even for representations that are designed to be resistant to various transformations in the visual appearance of objects in the cases tested. We then described our multi-level representation based on conceptual spaces, allowing us to define and explore heterogeneity between robots at multiple levels. In this framework, raw sensory data is abstracted into an intermediate representation called *properties*. These properties are represented as Gaussian mixture models, and can be learned via supervised learning. The algorithms and processes necessary for the learning of properties and concepts were described in this chapter. This representation was successfully used to learn a representation of objects and subsequently classify them, in both simulation and real-robot experiments.

In this chapter, we confirmed three hypotheses:

1. We demonstrated that the conceptual spaces representation can be used to learn underlying object properties as well as combine them to model objects themselves. We hypothesized successful classification (better than chance) as well as successful learning (improvement as the number of instances increases). Both of these were shown to occur in simulation as well as real-robot experiments.

2. We demonstrated that there is indeed a learning advantage to abstracting raw sensory data into the intermediate property representation in the cases tested. (Kira, 2010) This was shown using a state of the art machine learning algorithm (support vector machines). We hypothesized that learning curves, plotting classification accuracy as the number of training instances increases, would be higher when using the property abstraction compared to raw sensory data for learning. This was confirmed in a real-robot experiment by measuring the areas under the learning curves. In Chapter 5, we will show that in addition to this learning advantage, property abstractions can also facilitate transfer.

3. The final hypothesis we confirmed in this chapter is that perceptual heterogeneity prevents properties from being transferred directly between robots (Kira, 2009a). We transferred properties between two real robots, and showed that classification of property categories significantly decreased when a robot used properties received from the other robot compared to the properties it learned by itself.

This latter result motivates our contention that robots should explicitly learn about and model their differences in order to facilitate knowledge transfer. This leads us to the next chapter, where we present methods and robot interactions for learning which properties on one robot correspond to properties on another robot, even if the underlying spaces for these properties differ between the two robots.

CHAPTER 4

BUILDING MODELS OF PROPERTY AND CONCEPTUAL DIFFERENCES ACROSS ROBOTS

4.1. Sources of Heterogeneity between Two Robots

Robots can differ at multiple levels and in different ways. The way in which two robots differ impacts the efficacy of knowledge transfer between them. In this dissertation, we proposed to bridge the gap between two robots at the lowest sensory level by building an intermediate property representation. We showed that such abstraction can aid learning in Chapter 3, and will show that it can successfully bridge the gap between two robots that use different underlying raw sensory data during knowledge transfer in Chapter 5. Given this, the question then becomes:

How can two robots determine which properties they have in common?

This chapter focuses on this question.

Furthermore, some properties are more important for the representation of one concept versus another. If properties that are important to a concept are not shared, it does not matter that all of the other properties a robot has *are* shared. We deal with this issue in this chapter as well. We will demonstrate that robots can indeed determine which properties are potentially shared. The process involves joint interaction between the two robots in the same environment, leveraging the fact that the robots are embodied. Once property mappings are learned, they can be used to transform concepts when transfer occurs from one robot to the other. Furthermore, by knowing which properties are shared and which are not, in addition to knowing how important a set of properties are to representing a concept, the information lost by going from one robot to another can be measured via a metric. We will demonstrate this metric in Chapter 6. Note that in this and the next chapter, we focus on heterogeneity classes H2a and H2d, where properties are either shared or unshared. We will cover other heterogeneity types, where properties can overlap by different degrees, in Chapter 6.

4.2. Modeling Differences in Properties

As mentioned, properties are regions in domains, in our case represented as Gaussian mixture model clusters. The same property can be represented as clusters with different characteristics (for example, different standard deviations) or even domains from different sensors (for example, the width of an object as detected by a camera or laser). Given these clusterings of a domain, the problem is to find associations between clusters from each robot. In other words, the robots determine which cluster(s) in one robot belongs to which cluster(s) in another robot.

In order to do this, we use an interactive process in which the robots view the same scene and compare properties that they see. Given a scene, each robot processes its sensory data to produce a set of knoxels where property memberships in relevant domains can be calculated. For each pair of properties (one from each robot), the statistics described below are maintained in order to determine whether they represent similar physical properties. Note that using this method, the robots can also determine whether the robots differ in the properties that they can see due to differences in perspectives. This is important to determine, as shown in the first experiment conducted in Chapter 3.

4.2.1. Confusion Matrices

The problem of finding mappings between clusters is closely related to comparing different *clusterings*, which has been dealt with in statistics and machine learning communities (Rand, 1971),(Fowlkes and Mallows, 1983),(Meila, 2002). This line of research attempts to create measures of similarity between two clusterings. A major representation used in the creation of some metrics is the *confusion matrix*, which is a matrix with k rows (one for each cluster in the first clustering) and k' columns (one for each cluster in the second clustering). Each entry contains the number of points that belong to the cluster represented by the row (in the first clustering) *and* that belong to the cluster represented by the column (in the second clustering). In other words, it is the intersection of the clusters C_k and C'_k. For the problem of comparing clusterings, the confusion matrix is used to calculate some metrics for comparing different clusterings.

In our case, we seek to map *individual* clusters to each other, and not to determine overall similarity between the clusterings of the entire space. Hence, instead of calculating such a comparison, we utilize the confusion matrix to determine pairs of properties that may *potentially* represent the same physical property. Suppose that there are two clusterings G_i^A and G_j^B defining regions corresponding to properties p_i^A and p_j^B for robot A and B, respectively. Also, each clustering for robot A and B has n_j^B and n_j^B clusters, respectively. Finally, suppose that we have a set of instances I^A and I^B from each robot (obtained using its own sensing) with a sufficiently high membership defined by a threshold for property p_i^A. Specifically, the property confusion matrix $PC^{A,B}$ is calculated as:

$$PC^{A,B}_{(j,k)} = \frac{1}{|I|}\sum_i \frac{\min(s(i,p^A_j), s(i,p^B_k))}{s(i,p^A_j)} \qquad \forall j,k \qquad \textbf{(7)}$$

This matrix is very similar to the concept matrix used to represent concepts (described in subsection 3.5), except that it represents properties from *different* spaces along the rows and across columns, and is used to infer which properties map to each other. Again, the *min* function is used to represent the intersection of property memberships. For each property of a robot, the highest values in the corresponding property's row will be taken and it will be considered potentially corresponding to the respective property of the other robot. This is only true if the value is above a threshold, however; otherwise, it is considered not to have a corresponding property in the other robot's representation.

Table 16 summarizes the algorithm. Figure 45 shows the process of updating one cell in

Figure 45 –Example demonstrating the update of one cell in the confusion matrix, based on one instance. This is done for all instances, and between all properties of robot A that have membership above a threshold and all properties of robot B.

Instances from robot A:

$s(i^A, p^A_j)$	p^A_1	p^A_2	p^A_3	p^A_4	p^A_5
i^A_1	0.0	0.00	0.10	$\boxed{0.86}$	0.08
i^A_2	0.00	0.09	0.49	0.00	0.04
i^A_3	0.00	0.61	0.00	0.00	0.60
…					

Instances from robot B:

$s(i^B, p^B_j)$	p^B_1	p^B_2	p^B_3	p^B_4	p^B_5
i^B_1	0.00	$\boxed{0.75}$	0.10	0.04	0.08
i^B_2	0.00	0.03	0.09	0.00	0.60
i^B_3	0.00	0.02	0.85	0.00	0.60
…					

j = 4, k = 2, |I|=30,

$PC^{A,B}_{(4,2)}$ before update is 0.261

$$PC^{A,B}_{(4,2)} = PC^{A,B}_{(4,2)} + \frac{1}{I}\frac{\min(0.86,0.75)}{0.86}$$

$$= PC^{A,B}_{(4,2)} + \left(\frac{1}{30}\right)0.87 = 0.29$$

$PC^{A,B}$

	p^B_1	p^B_2	p^B_3	p^B_4	p^B_5
p^A_1	0.48	0.00	0.00	0.00	0.05
p^A_2	0.00	0.09	0.49	0.00	0.04
p^A_3	0.00	0.08	0.00	0.00	0.60
p^A_4	0.10	$\boxed{0.29}$	0.00	0.09	0.16
p^A_5	0.17	0.31	0.00	0.43	0.07

Table 15 - Learned Confusion Matrix $PC^{B,B}$. Each value measure the correlation between pairs of properties, with higher values indicating higher correlations. Values in the diagonal are all 1.0, since all properties are self-correlated. However, other properties can have non-zero correlation as well depending on the sensors and set of concepts used.

	p_1^B	p_2^B	p_3^B	p_4^B	p_5^B
p_1^B	**1.00**	0.02	0.00	0.01	0.05
p_2^B	0.00	**1.00**	0.05	0.03	0.23
p_3^B	0.00	0.07	**1.00**	0.00	0.04
p_4^B	0.00	0.22	0.00	**1.00**	0.01
p_5^B	0.05	0.00	0.00	0.00	**1.00**

the confusion matrix.

This example uses the first five properties in the color domain shown in the previous chapter. Note that the two matrices $PC^{A,B}$ and $PC^{B,A}$ may differ, since the first robot decides which instances to use to update a particular property based on whether its memberships is above a threshold. The mappings between properties of robot A and properties of robot B can be inferred by taking the maximal values in each row, with an optional empirically-determined threshold.

In some cases, there are other values in the same row that are relatively high. Some of this can be attributed to correlations between properties on the *same* robot. To see this, Table 15 shows the confusion matrix between all properties for the *same* robot (robot B). Besides having a maximal value in the diagonal of the matrix (since all property values correlate with themselves), there are additional high values. For example, when p_4^B had a large membership, p_2^B did as well. This is likely because correlations exist between the properties either because they are overlapping, or because of correlations in the training data (for example, brown apples correlates with rough texture). In order to remove some of these correlations specific to individual robots, the two confusion matrices $PC^{A,B}$ and $PC^{B,A}$ can be combined by an element-wise multiplication of one by the transpose of the other. Figure 46 shows an example. The resulting matrix, after normalization so that each row sums to one, differentiates the mapped properties with a reduced effect of inter-property correlations. Table 16 summarizes the algorithm.

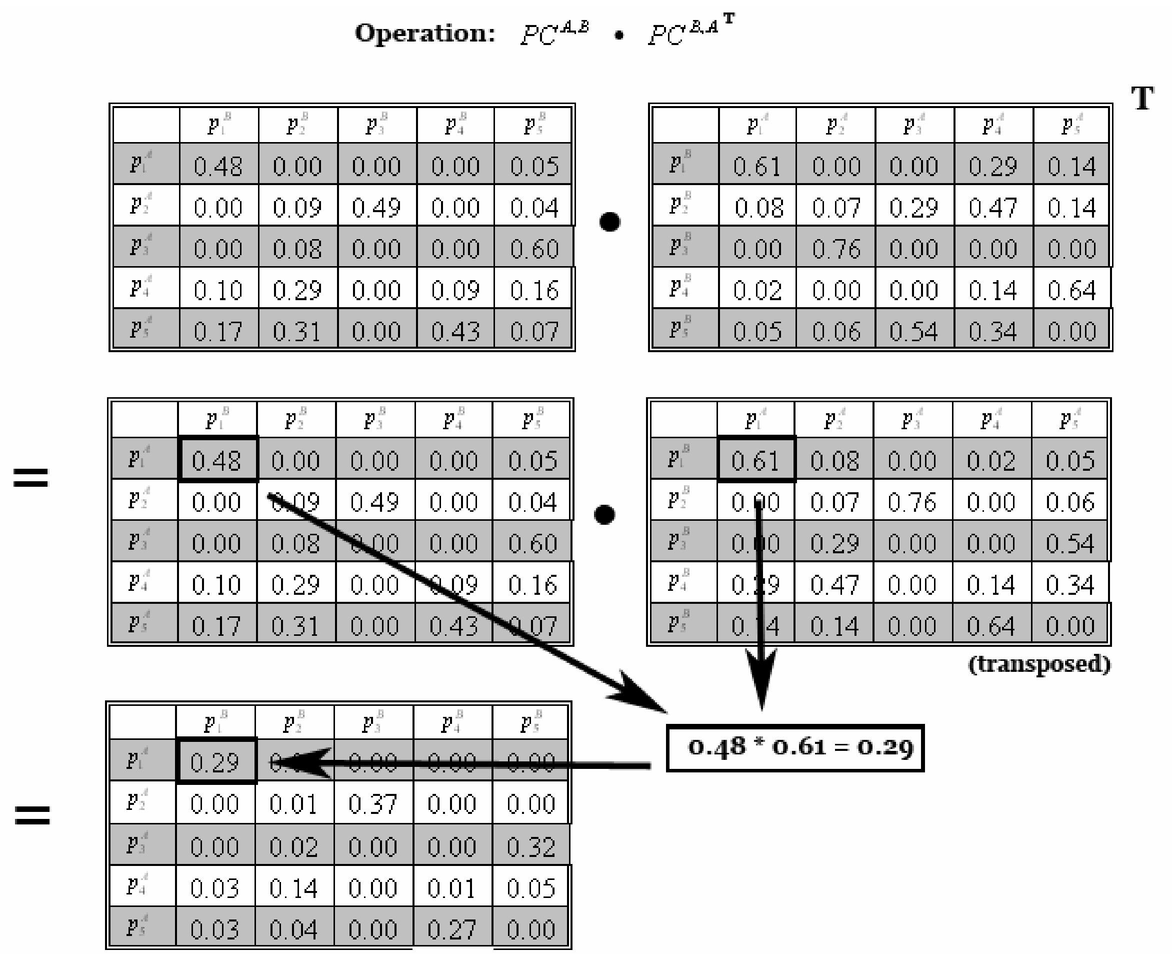


Figure 46 – Example of element-wise multiplication of the confusion matrix from one robot and transposed confusion matrix from another robot. This removes inter-object property correlations.

Table 16 – Algorithm for building the confusion matrix from data.

Algorithm: Building the Confusion Matrix from Data

Input: Properties *P1*, *P2* (from robot 1 & 2)
Output: Confusion Matrix CM
 	
// Establish a shared context. This can be done using manually-picked images
// viewing the same scene, or behaviors such as following, pointing, etc., e.g.
// as described in (Kira & Long, 2007).
 	
For each shared context instance *si*
 	
    // Calculate property memberships for both robots
    Values1 = *P1*(*si*)
    Values2 = *P2*(*si*)
 	
    // Add them to training instances
    Instances.add(*si,* Values1, Values2)
End
 	
For each property *p1* in *P1*
  For each property *p1* in *P2*
    // Make sure there is low uncertainty; that is a high membership for *P1*
    GoodInstances = Find in Instances values where *p1* is maximal
 	
     CM[P1][P2] = 0
     For each GoodInstance *gi*
       AddedValue = min (*gi*.Values1(*p1*) , *gi*.Values2(*p2*)) **(equation 7)**
       AddedValue = AddedValue / *gi*.Values1(*p1*)
       CM[P1][P2] = CM[P1][P2] + AddedValue
     End
 	
     // Normalize by the number of instances
     CM[P1][P2] = CM[P1][P2] / GoodInstances.size()
  End
End
 	
Return CM

4.3. Modeling Differences in Concepts

Once shared properties are known, the robots can use a similar methodology to create a model of shared concepts. Specifically, given instances from two concepts c_i^A and c_j^B, a concept confusion matrix CC is calculated as follows:

$$CC_{(j,k)} = \sum_i \frac{\min(s(i,c_j^A), s(i,c_k^B))}{s(i,c_j^A)} \qquad \forall j,k \tag{8}$$

Again, for each concept we find the most similar ones and estimate information loss between the two concepts. In this case, a concept consists of multiple cluster regions in different domains, and so we combine the information loss within each domain. Note that this not only determines whether a concept is shared (i.e. they both have the representation for the concept), but by placing a threshold on the information loss it determines whether the concept *can* ever be successfully shared, assuming the current set of shared properties. It can even inform the robots which properties would be useful to transfer, by calculating the most informative property that would lessen the information loss.

4.4. Locally-Shared Context

In the context of machine learning and statistics literature discussed above, the clusterings that are being compared are always in the same space. In other words, they both utilize the same data, and the data uses the same dimensions. In our case, we are attempting to compare clusters *between* spaces, where the axes (the dimensions) may differ. Hence, there is an additional correspondence problem in terms of whether an instance in one space corresponds to the same instance in another space. Data gathered in

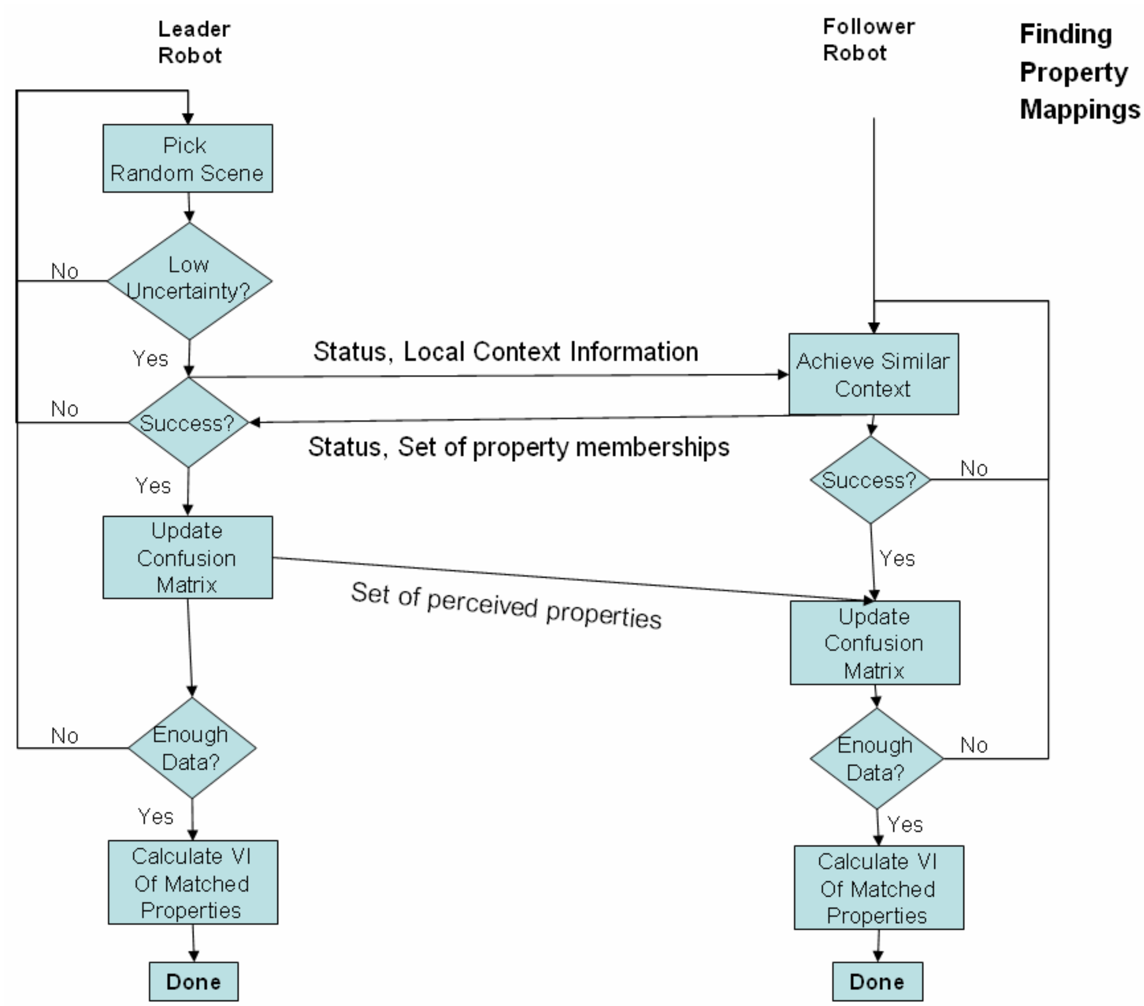


Figure 47 – Protocol for building models of property mappings between two robots. This same overall procedure is also used for building of concept mappings.

a random context from each robot will be difficult to analyze because of confounding variables such as differences in the environment or perspective. Hence, some shared context must be established first. This can be established using interaction such as following behaviors, etc. that are perceptually driven (assuming robots can detect each other, and determine things such as pose). Instances from each robot viewing the same scene can also be picked manually by a user or obtained by teleoperating the robots accordingly.

Alternatively, if they share certain properties that can provide context (e.g. shared frame of reference or other properties such as distances to objects, etc.) then these can be used instead. We call properties, such as locations in the environment, *context properties*. The strongest sense of a shared context is a *physically shared context*, in which two robots occupy the same (or similar) positions in the environment.

Figure 47 shows the general protocol involved. One robot, the leader, picks a random sensory data instance either from memory of by moving to a location in the environment. If the properties detected are not of high certainty, then a new scene is picked. Otherwise the leader robot sends the local context information (e.g. location) so that the follower robot can obtain sensory data from a similar context. Again, this can be done using methods from previous work (Kira and Long, 2007), where the follower robot moves to a nearby location, or it can obtain instances from memory fitting the constraints if they exist. The experiments in Kira and Long, 2007) did not use conceptual spaces, but the behaviors for achieving a shared context are nonetheless applicable. If pairs of instances are chosen manually or teleoperated, this is not needed. The two robots now have pairs of sensory data (and corresponding properties and concepts activated), and the confusion matrices can be calculated as described previously in the algorithm in Table 16 located in Section 3.5. Additional analysis in the form of information-theoretic metrics (Variation of Information or VI) can also be performed. This is covered in Chapter 6.

4.5. Experimental Evaluation: Building Property Mappings

We now describe experiments validating that robots can indeed build confusion matrices representing correct property mappings between robots ((Kira, 2009a) and (Kira, 2009b)). Table 17 summarizes the experiment.

4.5.1. Hypothesis

The hypothesis is that robots can leverage a shared context in order to build accurate mappings between shared properties. Ground truth is known since in these experiments properties are trained using the same real-world objects for both robots. For example, the "blue" property was trained using the same blue objects in the for both robots. Hence, we can use the supervised learning model in order to obtain the ground truth mappings. Most of the analysis will be done on the real robot data (both real robot configurations 1 & 2), although we will show that the same results can be obtained in simulation.

Table 17 - Experimental summary for the experiment the building of property mappings.

Experiment 5: General Experiment Summary **Building Property Mappings**	
Purpose	To determine whether robots can use instances from a shared context in order to infer property mappings between the robots.
Experiment Type	Simulation, Real-robot (configurations 1 & 2)
Hypothesis	We hypothesize that using instances from a shared context, robots will be able to build confusion matrices and subsequently infer the mapping between properties on one robot and properties on the other robot.
Procedure	1. Train properties using labeled data. 2. Obtain instances from a shared context 3. Build confusion matrices from these instances (algorithm in Table 16) 4. Obtain property mappings by taking maximal property pairs for each row 5. Compare property mappings to ground truth.

	p_1^B	p_2^B	p_3^B	p_4^B	p_5^B
p_1^A	**0.48**	0.00	0.00	0.00	0.05
p_2^A	0.00	0.09	**0.49**	0.00	0.04
p_3^A	0.00	0.08	0.00	0.00	**0.60**
p_4^A	0.10	**0.29**	0.00	0.09	0.16
p_5^A	0.17	0.31	0.00	**0.43**	0.07

	p_1^A	p_2^A	p_3^A	p_4^A	p_5^A
p_1^B	**0.61**	0.00	0.00	0.29	0.14
p_2^B	0.08	0.07	0.29	**0.47**	0.14
p_3^B	0.00	**0.76**	0.00	0.00	0.00
p_4^B	0.02	0.00	0.00	0.14	**0.64**
p_5^B	0.05	0.06	**0.54**	0.34	0.00

	p_1^B	p_2^B	p_3^B	p_4^B	p_5^B
p_1^B	**1.00**	0.02	0.00	0.01	0.05
p_2^B	0.00	**1.00**	0.05	0.03	0.23
p_3^B	0.00	0.07	**1.00**	0.00	0.04
p_4^B	0.00	0.22	0.00	**1.00**	0.01
p_5^B	0.05	0.00	0.00	0.00	**1.00**

	p_1^B	p_2^B	p_3^B	p_4^B	p_5^B
p_1^A	**1.00**	0.00	0.00	0.00	0.00
p_2^A	0.00	0.03	**0.97**	0.00	0.00
p_3^A	0.00	0.06	0.00	0.00	**0.94**
p_4^A	0.13	**0.60**	0.00	0.04	0.22
p_5^A	0.09	0.12	0.00	**0.79**	0.00

Figure 48 – Upper Left: Learned Confusion Matrix $PC^{A,B}$ from robot A's perspective. Upper Right: Learned Confusion Matrix $PC^{B,A}$ from robot B's perspective. Lower Left: Learned Confusion Matrix $PC^{B,B}$. These confusion matrices represent correlations between properties from respective robots. Bold values are maximal values in the rows, while highlighted cells represent ground truth. All mappings were correctly obtained. The matrix on the lower right is the normalized combined confusion matrix. This matrix combines $PC^{A,B}$ and $PC^{B,A}$ and shows the result after normalization.

4.5.2. Real Robot Results (Configuration 1)

4.5.2.1. Procedure

In order to conduct experiments with regard to building property mappings, we first used the learned property data from the previous experiments for this configuration (Section 3.9). In all of these experiments, we used an RGB color space for robot A and HSV color space for robot B. 75 test images were used for learning the confusion matrices. For each training instance, images viewing the same object were paired and given as instances from a shared context. Given this data, the algorithm in Table 16 was performed. Specifically, the property memberships for both robots were calculated, and values in the confusion matrix corresponding to properties that were maximally high in each instance were used to update the cell.

4.5.2.2. Results

Figure 48 (upper left) shows the confusion matrix from robot A's perspective and Figure 48 (upper right) shows it from robot B's perspective. For intuition, each value $PC_{(j,k)}^{A,B}$ in the matrix is modified for each instance in which property j has the largest membership according to robot A's property models. The amount that it is updated by depends on the property membership ascribed to an instance in the same context by robot B (see equation 7 and Figure 27 for an example). Note that the two matrices may differ (as they do in this case), since the first robot decides which instances to use to update a particular property based on whether its memberships are the highest compared to the other properties.

As discussed, the inferred mapping between properties of robot A and properties of robot B can be determined by taking the maximal value in each row (in bold). In this case, the maximal values in both matrices (in bold) corresponded to the correct mappings (highlighted). This can be verified using Table 19, which shows the ground truth mapping; for example, in $PC^{A,B}$ the highest value for row p_2^A is in the column corresp-

Table 18 – Left: Numerical representation of un-normalized confusion matrix for the five color properties. Right: Numerical representation of confusion matrix for the three texture properties. Bold represents maximal values in rows. Highlighted cells represent ground truth mappings. All mappings were correctly obtained, as can be verified from Table 19.

	p_1^A	p_2^A	p_3^A	p_4^A	p_5^A
p_1^B	**0.29**	0.00	0.00	0.00	0.00
p_2^B	0.00	0.01	**0.37**	0.00	0.00
p_3^B	0.00	0.02	0.00	0.00	**0.32**
p_4^B	0.03	**0.14**	0.00	0.01	0.05
p_5^B	0.03	0.04	0.00	**0.27**	0.00

	p_1^A	p_2^A	p_3^A
p_1^B	0.03	0.00	**0.35**
p_2^B	**0.41**	0.09	0.03
p_3^B	0.19	**0.24**	0.00

Table 19 – This table shows the color properties trained (e.g. "blue") and the corresponding property number for each robot. The property numbers represents the ground truth mappings between the robots and can be used to verify the property mappings learned by the robots.

	Brown Objects	Black Objects	Blue Objects	Gray Objects	White Objects
Symbol: Robot A	p_1^A	p_2^A	p_3^A	p_4^A	p_5^A
Symbol: Robot B	p_1^B	p_3^B	p_5^B	p_2^B	p_4^B

onding to p_3^B (0.49), which is correct. In some cases, there are other values in the same row that are relatively high. Some of this can be attributed to correlations between properties on the same robot. For example, when p_4^B (corresponding to white) had a large membership, p_2^B (corresponding to gray) did as well (see Figure 48, lower left). This is because some gray objects were light gray and some white objects were dirty or not purely white. This indicates that independent properties are preferable for cross-robot mapping. If these correlations are segregated by combining both of the robot's learned confusion matrices, the resulting matrix differentiates the mapped properties more profoundly. This can be seen in Figure 48 (lower right). Similar results were obtained

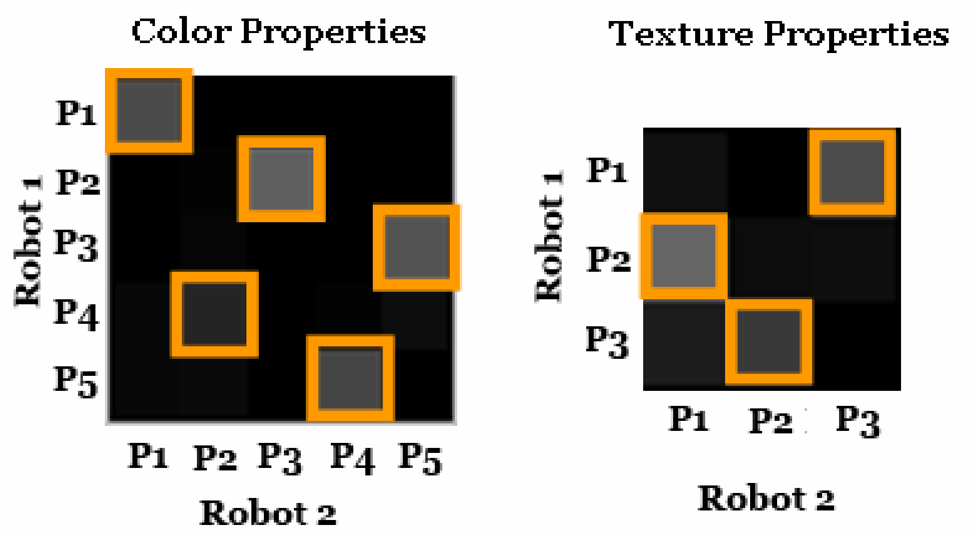


Figure 49 – Gray-scale representation of property mappings. Highlighted values indicate ground truth mappings.

for texture properties, where all of the correct mappings were inferred (again, this can be verified by comparing the maximal values in the matrix on the right in Figure 49 to the ground truth mapping in Table 19). Figure 49 shows the gray-scale representation for the color and texture mappings (in this case they were separated), and Table 18 shows the numerical values. Again, the mappings were determined with 100% accuracy.

4.5.3. Real Robot Results (Configuration 2)

We now describe results for determining the property mappings between the robots (Kira, 2010). Recall that in this configuration, color, texture, size, and shape properties were used. Color properties represented median RGB or HSV values for the object, while texture was represented using the mean and standard deviation of the output of a single empirically-derived Gabor filter over the entire object. Size and shape, which were only detected by the Pioneer robot, were determined using the lidar sensor. In order to learn the mappings, the confusion matrix was created using all test instances of the thirty-four

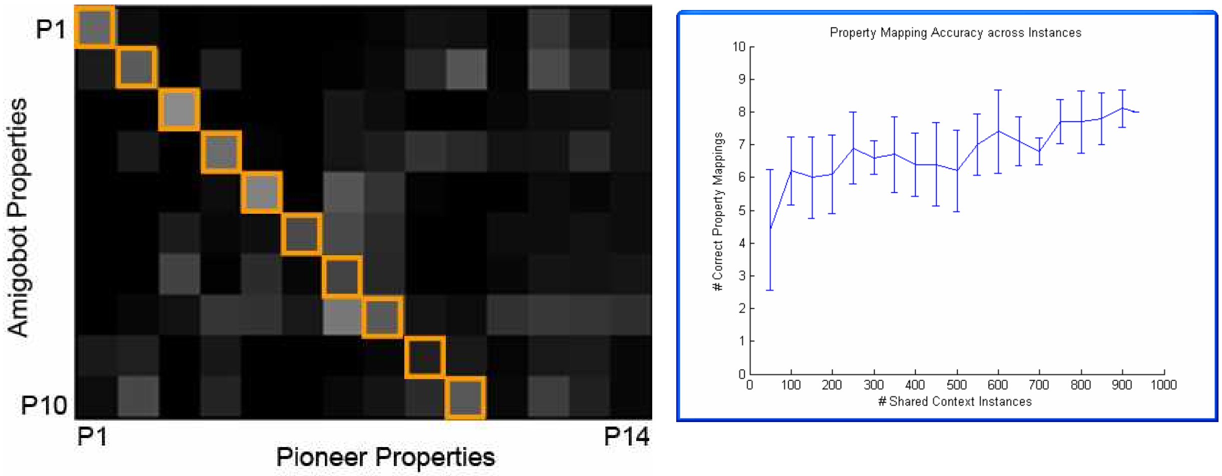
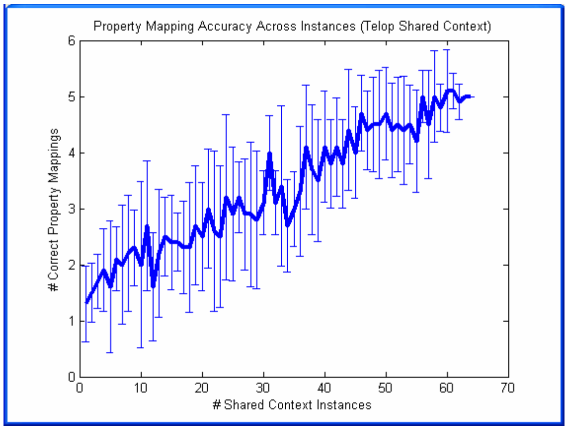


Figure 50 – Left: Gray-scale representation of the property mappings, where the rows correspond to properties of the Amigobot robot and columns correspond to properties of the Pioneer robot. Note that the latter robot has four more properties, utilizing its SICK range finder. By taking the maximal values of each row, eight of ten properties are mapped correctly (ground truth is the diagonal, highlighted). Right: This graph shows the number of correctly mapped properties between the robots as the number of instances grows (learning curve). For each point, the corresponding number of instances are used to build the confusion matrix, maximal values for each row are determined, and the mappings are compared to ground truth. The end of the graph is significantly different than the first point ($p<0.0001$) demonstrating significant learning.

objects. Hence, the shared context in this case was manually guaranteed (i.e. images from each robot sensing the same object were chosen). For each instance, each robot picked properties that were high for that instance, and added to the average the ratio of the other robot's property membership to its own.

Figure 50 (left) shows a gray-scale representation of the learned confusion matrix, where lighter values correspond to higher values (i.e. more highly correlated properties). The diagonal represents the ground truth mappings (since we trained the properties in the same order) and are highlighted. Note that there are fewer rows than columns since four of the properties do not exist on the second robot (heterogeneity type 2d). By taking the maximal values in each row, eight of ten properties were mapped correctly.

In this case, the texture properties were highly correlated across all objects, meaning that the properties were not independent. Again, this shows that such dependencies can



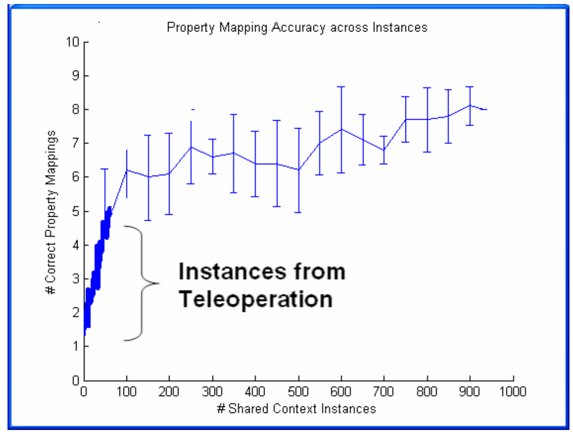


Figure 51 – Left: This graph shows the number of correctly mapped properties between the robots as the number of instances grows, using instances from the teleoperated robots. Right: This graph overlays the graph when using manually chosen images (thin blue line) with the graph on the left (bold blue line), showing that similar trends are obtained.

cause errors in the property mappings, an important fact for future work when we will work on unsupervised learning of the properties since some unsupervised algorithms do not guarantee this. Figure 51 (right) shows the number of correct mappings, averaged across ten randomized validation runs, as the number of testing instances increases. As can be seen, there is an initial steep learning curve where the first five or six properties are correctly mapped. The rest of the properties are more difficult to map and take additional instances, likely due to cross-property correlations in objects. We also tested this with sixty four instances where the two robots were teleoperated to view the same object. Figure 51 (left) shows the graph when using those instances. As can be seen, the color properties that were not correlated in the same object (unlike the texture properties) were quickly learned. After all of the instances were processed, five of ten property mappings were correct (when compared to the known ground truth), showing similar results as the graph where images were manually chosen (Figure 50, right). Figure 51 (right) combines the two graphs to show that similar trends are obtained in both situations. This shows that, assuming the robots can localize or detect each other, these

	p_1^B	p_2^B	p_3^B	p_4^B	p_5^B
p_1^A	0.000	**0.527**	0.000	0.001	0.374
p_2^A	0.000	0.000	**0.488**	0.085	0.182
p_3^A	**0.718**	0.000	0.001	0.070	0.017
p_4^A	0.139	0.051	0.109	**0.310**	0.160

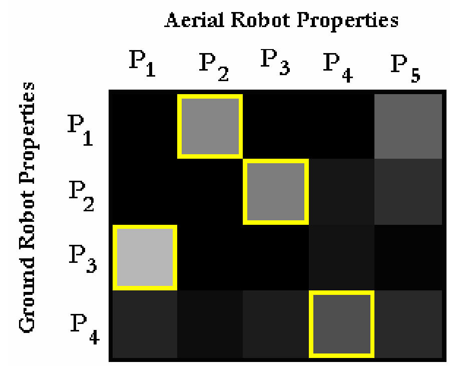


Figure 52 – Simulation results. Left: Property mappings in the form of a confusion matrix. Bold cells represent the maximal values along the rows. These also correspond to the ground-truth mappings, which are highlighted, showing that all correct mappings were found. Right: Gray-scale representation of property mappings between the ground and aerial robots, where larger values are lighter. Highlighted cells represent the ground truth mappings and also correspond to maximal values..

mappings could be learned autonomously once behaviors for following or pointing from our previous work performed in simulation are applied to the real robots (Kira and Long, 2007).

4.5.4. Simulation Experiments

We now demonstrate that the same mappings can be learned in simulation. In these experiments, manual selection of image pairs (one from each robot) containing the same object was performed. However, since one robot was on the ground while the other was in the air, the perspectives were different. Figure 52 shows the resulting matrix in gray-scale image format. The ground truth mappings are highlighted and can be verified from Table 8. The maximal values of each row correspond to the correct mapping, although there is an ambiguity between the first property of the ground robot ("brown") and the fifth texture property of the aerial robot. This can be resolved in this case since it is less than the maximal property in the same row, but future work will look into mechanisms for disambiguating such potential false positives. Table 20 summarizes the hypothesis and conclusions of this experiment.

Table 20 - Experimental conclusions for the experiment the building of property mappings.

Experiment 5: General Experiment Conclusions **Building Property Mappings**	
Hypothesis	We hypothesize that using instances from a shared context, robots will be able to build confusion matrices and subsequently infer the mapping between properties on one robot and properties on the other robot.
Conclusions	Hypothesis is confirmed. Property mappings were learned with a high degree of accuracy (100% accurately for simulation experiment and robot configuration 1, and 80% accuracy for robot configuration 2) using instances from a shared contex.

4.6. Summary

In this chapter, we have shown that two robots can use instances from a shared context in order to infer mappings between their respective properties. This was done using confusion matrices, which measured the correlation between property memberships for all property pairs. Maximal values in the rows of the confusion matrix could then be used to infer property mappings. We hypothesized that a shared context would enable this, and the hypothesis was confirmed using simulation experiments (Section 4.5.4) as well as two different real-robot experiments (Section 4.5.2 and Section 4.5.3). Given that robots can use property abstractions to successfully learn concepts (shown in Chapter 3) and that robots can map properties between each other (shown in this chapter), we now move on to demonstrate knowledge transfer given these property mappings. This is done in the next chapter.

CHAPTER 5

CONCEPT TRANSFER USING PROPERTY MAPPINGS

The preceding chapter required the two robots to be in the environment in order to build models that map shared properties and concepts. This was done in order to find out what *a priori* differences and similarities exist. We now describe various scenarios in which these models can be used, starting with the clear application of knowledge exchange in the form of one robot communicating an entire concept to another robot. Using the methods described previously, it can be detected whether a concept in one robot exists in the other robot's representation within an acceptable amount of information loss. If this is not the case, then the entire concept's representation can be communicated and assigned a new label by the receiving robot. It may be that the reason the concept does not exist in the second robot is that it does not have the proper dimensions or properties to describe the concept accurately; in this case, the process of transferring the concept will fail as well. However, it may be that the second robot simply did not encounter the concept but has sufficient capabilities to describe it, in which case the process would succeed.

5.1. Perceptual Heterogeneity: The Space of Possibilities

The richness of communication between two robots can vary depending on whether the two robots share similar dimensions, properties, and concepts. Table 21 shows a table enumerating all of the possible configurations. The first two situations involve two robots that do not share any similar properties. In other words, $P_c^{A,B}$ is the null set and additionally there are no shared context properties. In this case, regardless of whether the robots utilize a common representation, structured knowledge sharing (i.e. sharing concepts directly and their hierarchies) is not possible. Supervised learning, however, may be possible where one robot provides a label to the other. This is only true if they share some similar *dimensions*, whereby one robot can send its values to the other with a label, or they have various behavioral methods for achieving a locally shared context, such as those described in (Kira & Long, 2007). The receiving robot can then learn concept models using its own properties from this labeled data. This boils down to the second robot performing supervised learning of concepts from scratch. In general, it is unlikely that dimensions will be directly transferable though, as even sensors of the same models can produce dimension heterogeneity.

Table 21 – Types of Heterogeneity and Communication Possible

Similar Dimensions?	**Similar context properties?**	**Similar concept properties?**	**Type of Knowledge Alignment Methods Possible**		
			Entire Concept	**Context**	**Structure**
No	No	No			
Yes	No	No	✓		
No	Yes	No		✓	
Yes	Yes	No	✓	✓	
No	No	Yes			✓
Yes	No	Yes	✓		✓
No	Yes	Yes		✓	✓
Yes	Yes	Yes	✓	✓	✓

In most cases, there will be at least *some* properties in common. For example, even if visual properties are not shared, position information such as odometry may be. If only odometry is shared, then robots can establish a physically shared context and use supervised learning to train properties and symbols. This makes sense with respect to our notion of context, in that the only properties in common in this case are those that can be used to define a context (location properties). In general, if properties that can be successfully used to ensure a shared context are similar between the two robots (but the actual perceptual properties used to *describe the concept* are not), this type of alignment

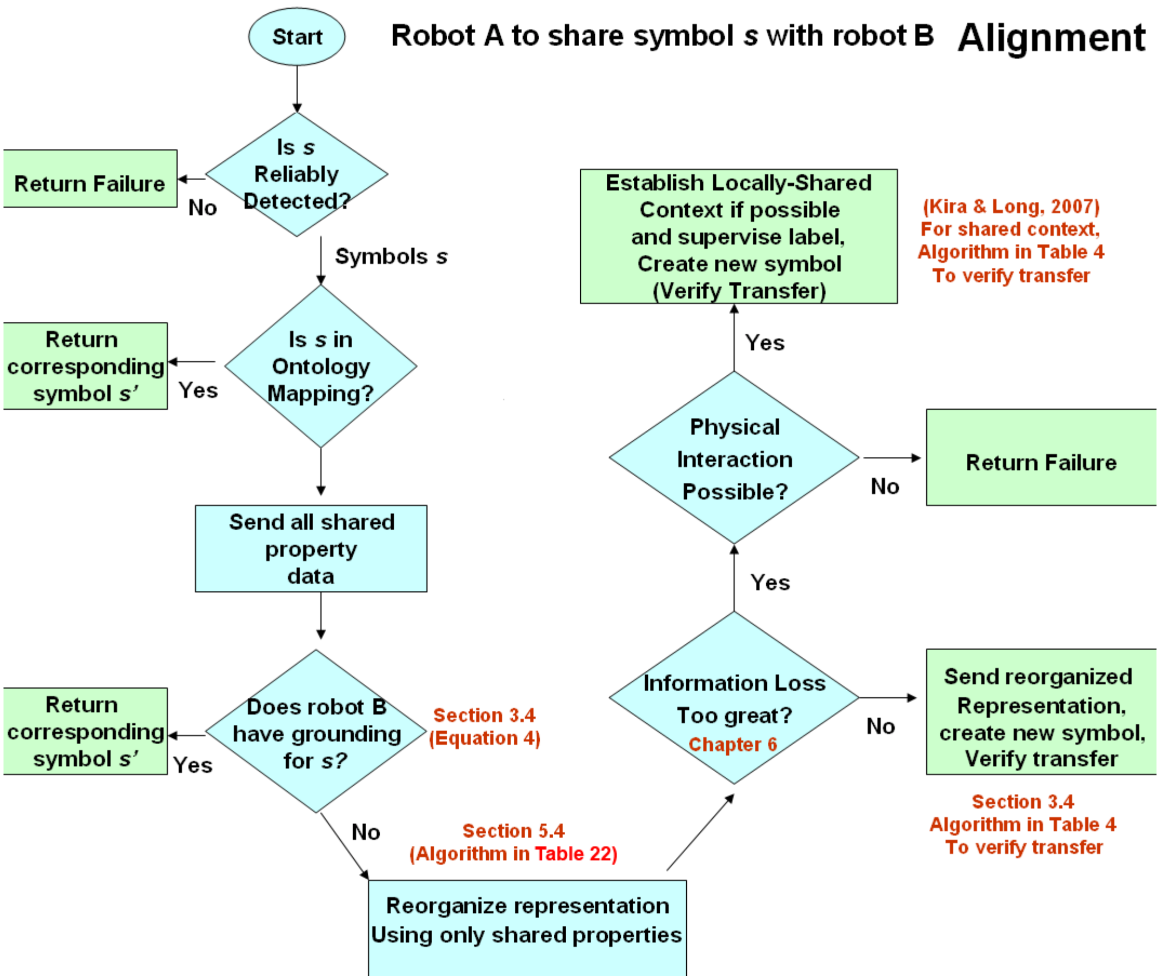


Figure 53 - Flow chart for alignment of a symbol (representing a concept).

can be used. This is similar to supervised learning of concepts in the previous situation (when *no* properties are in common) except that such learning can be done when no dimensions are shared since each robot obtains data from its own sensors (that satisfy the shared context).

In any case, since the perceptual features used to describe objects are not shared, structured knowledge sharing is again not possible.

5.2. Ontology Alignment with Shared Properties

The richest form of communication is possible when underlying concept properties are shared. Entire concept matrices can then be transferred directly. The flow chart for mapping or sharing a symbol (representing a concept), is shown in Figure 53.

The algorithm takes in all of the knowledge representations and similarity models described previously, as well as a symbol *s* that robot *A* has in its knowledge base. The algorithm first checks if the ontology mapping model has information on the symbol, and if the similarity is greater than a threshold. This indicates that the mapping between symbol *s* and another symbol *s'* in robot B's knowledge base has already occurred. If this is not true, the actual alignment process begins.

First, a list of shared properties used in the representation of the concept is found. This is done using the property mapping model (as discussed in Chapter 4). The algorithm then inputs this to a reorganization or knowledge adaptation function that zeroes out entries in the concept matrix associated with unshared properties. In the case dealt with in this subsection, in which a sufficient number of properties used in the shared concept's representation, the function returns the same or slightly modified representation. This function only returns successfully if the loss of information, as measured by variation of information, is sufficiently small. If this is the case, it is then a simple matter of transferring this representation to robot B, assigning it the same symbol name, and updating the concept mapping model to reflect this new mapping. If the adaptation function returns unsuccessfully, then again the robots must fall back on sharing instances with a shared context and utilizing supervised learning where robot *B* attempts to learn to identify the symbol with its own dimensions and properties. This is different than the instance-sharing in the paragraph above, since if the properties used in the representation of the communicated concept are shared, then a shared context is not necessary; all robot A has to do is send *its own* representation to robot B. Note that this is only possible if physical interaction is possible.

No matter how a concept is shared, whether via direct transfer or by one robot providing a supervisory signal, it is important to note that the knowledge transfer may fail. In the case of direct knowledge transfer, the models describing which properties are shared may be noisy or contain errors, and in the case of providing a supervisory symbol it may be that the receiving robot does not have the capability to represent the concept correctly. Hence, it is important to verify that the transferred concept can indeed be detected by the receiving robot, and that the information loss is not unacceptable. This can be done using the protocols and algorithms described in Chapter 4 (specifically 4.2.1 and algorithm in Table 16), whereby confusion matrices are created, and Chapter 6 where variation of information is measured.

5.3. Sources of Error in Concept Transfer

We have argued for the importance of being able to estimate the effectiveness of knowledge transfer *a priori*. The goal is to utilize the models of differences and similarities learned between robots in the previous chapter. In order to do this, we must first analyze common sources of error when transferring concepts from one robot to the other.

One source stems from heterogeneity type 2d and 3d; that is, properties may simply be missing on one robot when compared to another robot (see Section 3.6). However, some properties are more important to representing a concept than others, usually due to different modalities being more important. For example, to a human, sound properties may not be as useful to classify a fruit while taste will be. This can take into account not only the inherent importance of a modality or property to an object, but also the availability of the modality. For example, in our case smaller objects that reside below or above the SICK lasers have no observed size or shape properties because the objects are simply not in the robot's point of view.

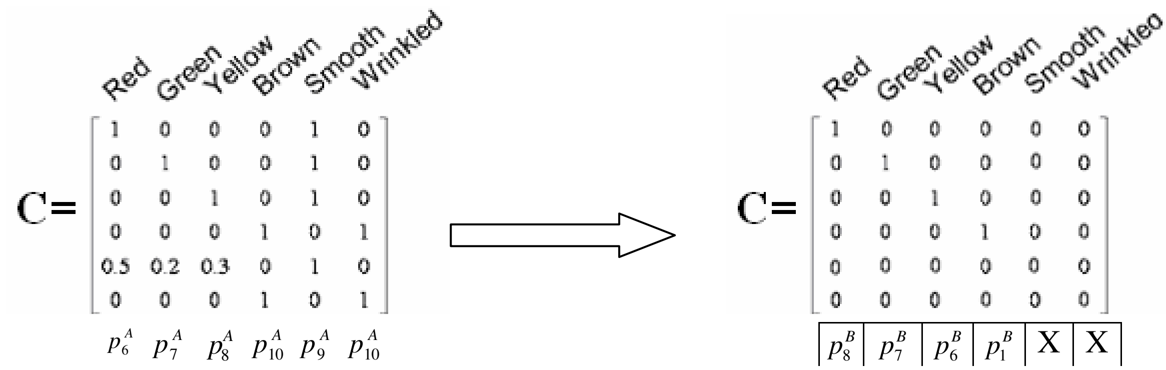
Since the confusion matrices described in the previous chapter encode precisely which properties are shared, these models can be used to estimate how effective knowledge transfer will be. Using test data containing known concept labels, the transferring robot can calculate how well a concept can be classified using the exact properties the receiving robot shares with it. Using the conceptual spaces representation, this is an easy process that can be done by zeroing out the appropriate rows and columns of the concept matrix, as was shown previously (Section 5.4).

A second source of error in transfer lies in heterogeneity types 2b and 3b, where properties are shared but not completely overlapping (see Section 3.6). In this case, it is more difficult for the sending robot to estimate the resulting performance on the receiving robot using a test set. However, in Chapter 6 we introduce an information-theoretic metric that will allow the characterization of the *amount* of overlap between properties. The robots will then be able to estimate information loss using this metric.

Note that in general, these methods only produce estimates of post-transfer performance. There can be additional sources of error such as random noise (for example, in the property memberships), poor recall rates of the properties, effects due to combinations of properties working in concert, etc. One strength shared by the conceptual spaces and this approach, though, is that if the receiving robot has *additional* properties, knowledge transfer will not negatively impact classification with respect to those properties. In other words, since the entries in the concept matrix for these additional properties will be zeroed out (because the properties are not shared), classification will not be affected. As we will show, however, the receiving robot can continue learning if it encounters the concept and learns the values for these unshared properties. Learning will be bootstrapped as much as possible by the shared properties by transferring shared properties, and learning can then continue on the receiving robot to additionally learn correlations between properties that it had but the other robot did not.

5.4. Transferring Concepts in Conceptual Spaces

We will now describe how concepts can be transferred from one robot to the other. Table 22 shows pseudocode for the overall algorithm. Suppose that robot A will attempt to transfer concept c to robot B. Let P_c^A be all of the properties involved in the concept, as defined previously in Section 3.3, Equation 2. Let P^B be all properties that robot B has. We define $P_c^{A,B} = P_c^A \cap P^B$ as the intersection of these sets, with two properties p_i^A and p_j^B being equal if $PC_{i,j}^{A,B}$, i.e. their confusion matrix value is smaller than a threshold (that is empirically determined). We then perform *knowledge adaptation*, where the connection matrix associated with concept c is transformed based on robot B's



Property Mapping Model

p_6^A	p_7^A	p_8^A	p_1^A	p_9^A	p_{10}^A
'p_A_6423'	'p_A_1826'	'p_A_1756'	'p_A_1289'		
p_8^B	p_7^B	p_6^B	p_9^B	X	X
'p_B_7831'	'p_B_1756'	'p_B_1876'	'p_B_8752'		

Figure 54 –Example demonstrating the transfer of a concept, using the property mapping model to remove unshared properties.

capabilities by zeroing out all values in row *i* and all values in column *j* for each property $p_i^B \notin P_c^{A,B}$.

Figure 54 shows an example. Upon receiving the concept matrix, robot B also substitutes its own property labels for each of robot A's property labels, based on the model mapping each robot's properties to each other. An interesting alternative to zeroing out values for non-shared properties is to create new labels and property numbers for robot B, tagging it with an attribute that specifies that it cannot be sensed by the robot, and leave the values in the matrix. This allows another form of knowledge transfer: the *augmentation* of new properties for a robot that it cannot actually sense. Using the values in the transmitted concept matrices, however, it will be able to assign a likelihood for the value of this property despite not being able to sense it. For example, it can infer that since an instance of an apple is brown, it is highly likely that it is wrinkled. In this dissertation we have not explored such augmentation, but it remains an interesting capability to explore for future work.

Table 22 - Algorithm for transferring a concept from one robot to the other and reorganizing the concept matrix based on shared and unshared properties.

```
Algorithm: Concept Transfer from Robot A to Robot B
Input: Properties P^A , Properties P^B , PropertyMapping PM, Concept matrix C^A
Output: Concept matrix C^B

// Note: PropertyMapping PM consists of an array where PM[ p ]
// contains the index to the property p' on robot A that maps to
// property p on robot B.  These mappings are obtained using the
// methods in Chapter 4.

C^B = Matrix of size |P^B|×|P^B| initialized with zeros
For each property P1 in P^B
  For each property P2 in P^B
    // Both properties P1 and P2 map to existing properties on robot A
    If ( PM[P1] > 0 & PM[P2] > 0 )
      // Find corresponding cell value based on property mapping
      C^B [P1][P2] = C^A [ PM[P1] ][ PM[P2] ]
    Else
      // One of the property pairs do not correspond to a property on Robot A, do
      // nothing
    Endif
  End
End

Return C^B
```

5.5. Experimental Evaluation Overview

Thus far, we have described a framework for ontology alignment between two robots. A key factor of the process is that it takes into account the levels at which two robots differ, which determines the amount and type of knowledge sharing possible. To evaluate the system, we wish to show that the model of differences learned previously and the alignment process described here can allow effective knowledge sharing. First, we provide evidence for our hypothesis that property abstractions aid knowledge transfer when the underlying representations for the properties differ. We then show that concept transfer using conceptual spaces is effective and results in better overall performance throughout the robot's learning process. We also analyze different situations in which different subsets of properties are shared between the robots to demonstrate the estimation of performance on the receiving robot after transfer. Most of the analysis will be done on the real robot data, although we will show that similar results can be obtained in simulation.

5.6. Experimental Results: The Importance of Property Abstractions for Transfer

We will begin our experiments by demonstrating that the property abstractions aid transfer, in addition to learning (which was shown in Section 3.8.4) (Kira, 2010).

5.6.1. Hypothesis

We hypothesize that property abstractions do indeed aid transfer in the case of different representations. Specifically, we hypothesize that transfer learning in this case will remain as effective when using properties but not when using raw sensory data.

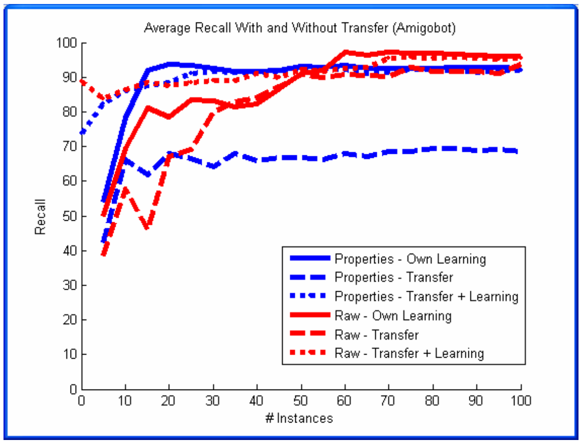
5.6.2. Procedure

These experiments use the same robots from configuration 2 and the procedure is the same as well (Section 3.10). Specifically, a Mobile Robots Amigobot with a wireless camera and a Pioneer 2DX robot with a Quickcam Express web camera were used**.** Properties were learned using algorithms in Section 3.5 (specifically the algorithm in Table 5). These learned properties were then combined to learn object models. For the condition using properties, each training instance was used to calculate the property memberships and these served as the input to the classifier. Again, we used support vector machines to classify objects. The input to the classifier was either the raw median RGB values or the property memberships, depending on the condition. Recall that two conditions were used in the experiment. In the first condition, the property memberships for the previously learned properties were used as attributes. In the second condition, raw sensory data itself (e.g. RGB values or the curvature metric) were used as attributes for training. In this section, unlike before, we also used a third condition where raw sensory data was used, but the robots used different representations for color. Namely one robot used an RGB color space while the other used an HSV color space.

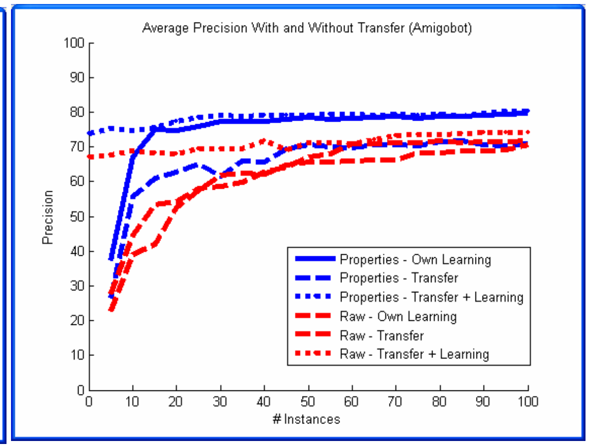
In order to gauge classification rates, both recall and precision are plotted as the number of training instances increases. These are standard classification metrics, where recall measures the number of true positives divided by the number of positives in the test set, while precision measures the number of true positives divided by the number of test set instances that were classified to be positive. The learning curves for both of these metrics were then plotted, showing the recall and precision rates as the number of training instances increased.

5.6.3. Results

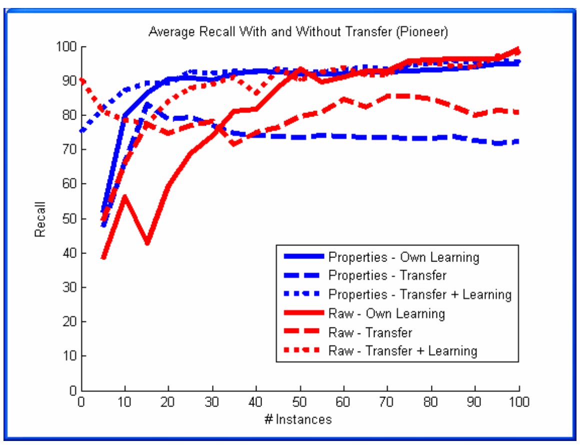
Figure 55 shows the recall and precision results when comparing the first two experimental conditions when the Amigobot is receiving learned representations from the Pioneer. As described in Section 3.10, the recall rate measures the number of true pos-



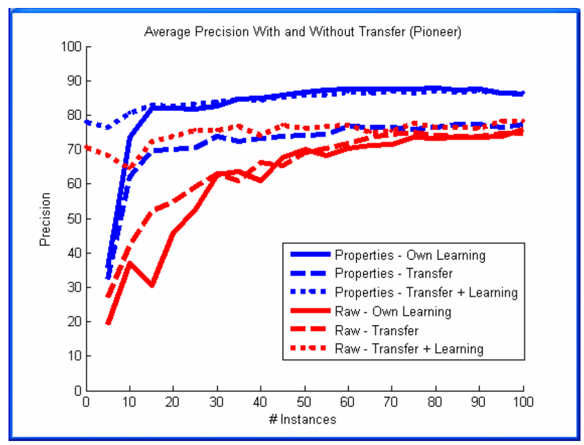
(a) Recall learning curve for the Amigobot. The areas under the curve were 0.86 when using properties and 0.84 when using raw values.



(b) Precision learning curve for the Amigobot. The areas under the curve were 0.72 when using properties and 0.61 when using raw values.



(c) Recall learning curve for the Pioneer. The areas under the curve were 0.86 when using properties and 0.78 when using raw values.



(d) Precision learning curve for the Pioneer. The areas under the curve were 0.80 when using properties and 0.59 when using raw values.

Figure 55 – Results demonstrating the advantage of using abstracted properties as opposed to raw sensory data when learing. The figures on the left show precision, while the figures on the right show recall. The figures on the top show results for the Amigobot robot, while the figures on the bottom show results for the Pioneer 2DX robot.

itives divided by the number of positives in the test set, while precision measures the number of true positives divided by the number of test set instances that were classified to be positive. These numbers are proportions and can be represented as percentages. The graphs compare results without transfer ("Own Learning"), transfer ("Transfer"), and continued learning by the receiving robot after transfer ("Transfer + Learning"). The "Transfer" curve shows how the concepts transfer as the expert robot learns on more instances. In other words, for each point in the curve (say x=10, i.e., ten instances), the expert trains on ten instances per concept and then transfers all the concepts to the receiving robot. The recall and precision rates on the receiving robot's test set (using the transferred representation only) is then calculated. This represents one point on the curve. For the "Transfer + Learning" curve, transfer occurs for all concepts after the expert robot trained on *all* of its training instances. The receiving robot then takes the transferred representation, continues to learn, and again tests the recall and precision rates on a test set. In that case, a point on the curve (say x=10, i.e. ten instances) means that the receiving robot took the transferred representation, continued adapting it using ten of *its own* training instances, and then tested its accuracy.

The blue curve shows results for classification using property memberships while the red curve shows classification using raw sensory data. As can be seen, transfer learning results in the bootstrapping of both recall and especially precision. This can be seen by the fact that the "Transfer + Learning" begins and continues *higher* than the "Own Learning" curve. In fact, the receiving Amigobot robot immediately classifies objects at an average recall rate of 74.0% and precision of 73.7%, *without having been trained on any instances itself*. The rates are significantly better than a classifier performing at chance rates (50%) and outperform rates achieved by the robot after five instances *when the robot learns by itself* (recall of 53.8% and precision of 37.2%). In other words, transfer learning improves upon learning with five instances by 37.5% for recall and 98.1% for precision. This is true despite the fact that the robot did not perform any training by itself. These results are the first demonstration in this dissertation of the clear advantage of knowledge transfer between robots.

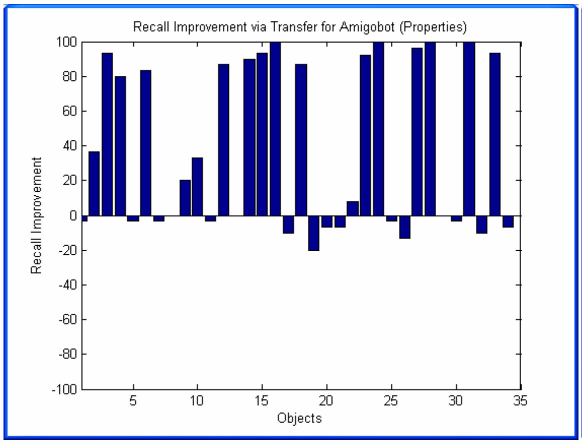
In the results above, the classification accuracy immediately after transfer was significantly better than chance and better than classification rates when the robot learns by itself after only five instances. This occurred not only for learning using properties, but was even more pronounced when learning with raw values (a more difficult task). Specifically immediate rates after transfer were 93.8% for recall and 70.6% for precision, compared to when the robot learns by itself after five instances where the recall was 50.0% and precision was 27.8%. These numbers represent an 87.6% improvement in recall and 154.0% improvement in precision. This shows that as learning becomes more difficult, *transfer learning becomes even more advantageous*. This is because lower rates are achieved by the robot when learning by itself (for harder learning tasks), but transfer learning is still effective.

Note that the transfer graph in Figure 55 represents the transferred SVM classifier being directly tested on testing data from the receiving robot. To perform continued learning after receiving classifiers from another robot, we instead use the support vectors from the transferred SVM classifier as input instances to a new classifier. This sometimes led to a slight performance change (e.g. an increase of 2.84% for recall and 5.26% for precision when using properties, and a decrease of 5.11% for recall and 3.53%

for precision when using raw sensory data). Subsequently, additional training instances were added as input to the classifier (plotted as "Transfer + Learning"). As can be seen from the "Transfer + Learning" curves, the Amigobot robot could achieve higher recall and precision (74.0% and 73.7%, respectively), even without having seen any instances by itself, compared with rates after training by itself with only five instances (e.g. 53.8% and 37.2%, respectively).

As the receiving robot began to receive additional training instances, it could combine the received classifier with these instances and eventually achieve similar rates than when learning by itself from scratch (recall of 92.1% and precision of 80.3% after all instances in the "Transfer+Learning" condition, compared to 92.7% recall and 79.6% precision after all instances in the "Own Learning" condition). *This shows that combining learned knowledge with received knowledge did not pose a problem in this case*. The same trends exist for the learning curves of the Pioneer robot (recall of 96.3% and precision of 86.1% after all instances in the "Transfer+Learning" condition, compared to 94.8% recall and 86.3% precision after all instances in the "Own Learning" condition).

One aspect of knowledge transfer between robots is that certain concepts may be transferred more effectively than others. This was seen in our first experiment using SIFT vocabulary trees (Section 3.1), where catastrophic failures occurred for some objects. Figure 56 shows a bar graph of the difference between using transfer learning (and no continued learning) over learning by the receiving robot itself after only five instances. In other words, for each object, we subtracted the recall and precision rates when the robot learned by itself for five instances from the recall and precision rates after



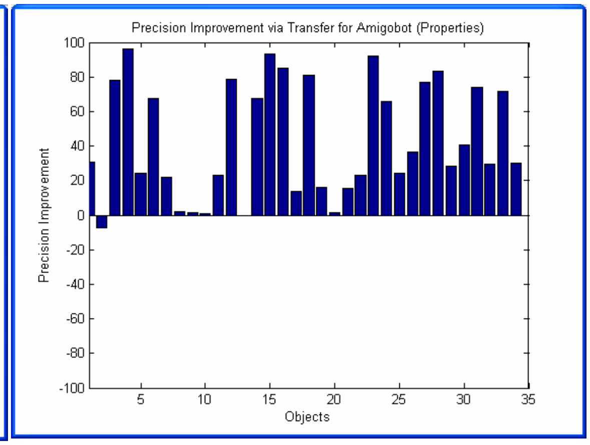


Figure 56 – Bar graph showing recall (left) and precision (right) improvements when using knowledge transfer compared to learning after only five instances. Positive values indicates an improvement when using transfer learning. 62% of objects (21/34) received an improvement in recall and the for the rest the rates were never worse than a 20% decrease. 97% (33/34) of objects received an improvement in precision.

transfer learning. The differences are shown for all thirty-four objects. Positive values indicate an improvement in the rates when using transfer learning over self-learning with five instances. As can be seen, the large majority of objects had positive improvement, and only a few objects had negative degradations in recall (all decreases were less than 20%) and only one object had a negative degradation in precision (a difference of less than 10%). In one sense, this is an unfair comparison, as transfer is compared to learning after five instances. Transfer is an improvement across the board with *no* training instances at all, as the results without transfer would correspond to random chance.

The results shown thus far depict results when both robots use the same underlying color space (RGB) for the color properties. In that case, transfer learning was successful both when using properties or raw sensory data as discussed above. This is a bit of a surprise, as raw RGB values do differ between the two robots for the properties.

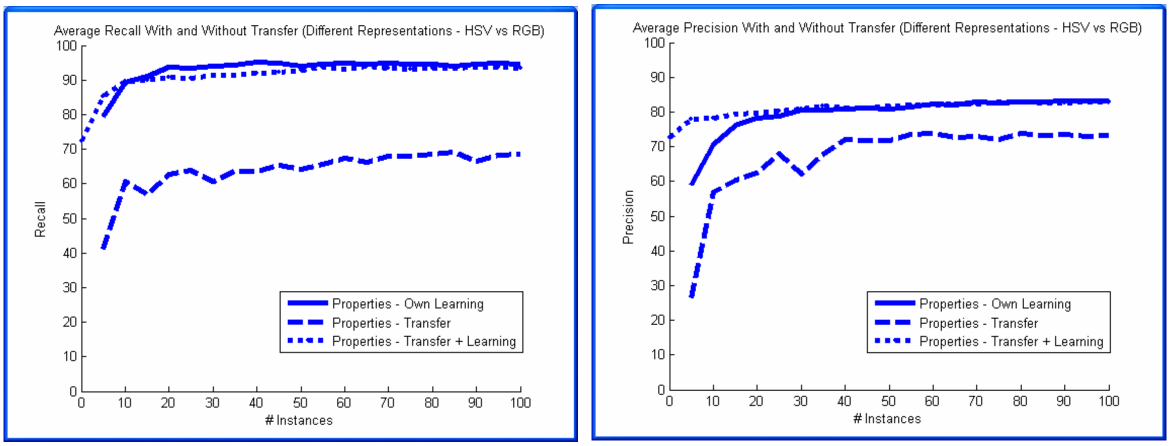


Figure 57 – Results demonstrating the success of transfer learning when using properties, even when the underlying representations used by the robots differ (one uses an RGB color space while the other uses HSV). This data shows classification by the Amigobot robot. The "Transfer + Learning" are higher than the "Own Learning" condition (

However, it seems that the discriminative learning (support vector machines) is powerful enough to overcome these differences in distributions by normalizing the attributes.

We now test our hypothesis that knowledge transfer using properties will continue to be successful even when the properties themselves are represented using different spaces on each robot. In this case, the Amigobot robot used the HSV color space to represent the same color properties. Other properties, such as texture, remained the same as before (and the same as the other robot). Figure 57 shows the results when using the abstraction of data via properties. Despite the fact that the robots used different underlying metric spaces, transfer learning was still successful. The advantage is less pronounced for recall in the transfer case when the two robots used different representations compared to when both robots used RGB, but overall transfer learning allowed the receiving robot to classify instances well, especially in terms of precision.

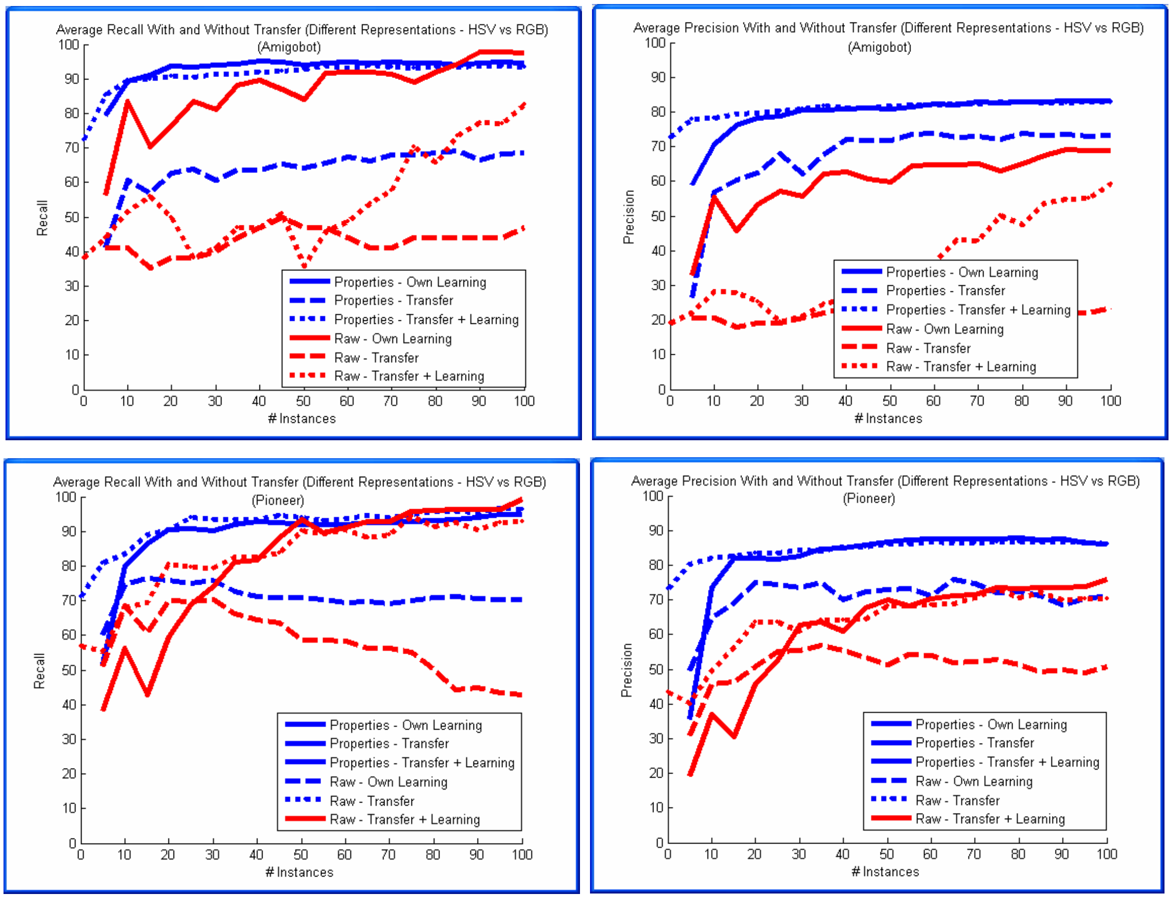


Figure 58 – Results demonstrating the success of transfer learning when using properties, even when the underlying representations used by the robots differ (one uses an RGB color space while the other uses HSV). This data shows classification by the Amigobot robot.

Figure 58 show the classification results in Figure 57 (in blue), when compared to using raw sensory values (red). The results are shown for both robots. As can be seen, transfer learning when using properties to learn continued the same trend as before, with gains after transfer over self-learning after five instances even when the receiving robot had not seen any learning instances itself (recall of 72.2% and precision of 72.8% immediately after transfer, compared to 79.3% recall and 59.0% precision after five instances in the "Own Learning" condition). In this case the recall immediately after transfer is lower compared to learning with five instances but the overall area under the learning curves is still higher for transfer learning as we will see below (Figure 60).

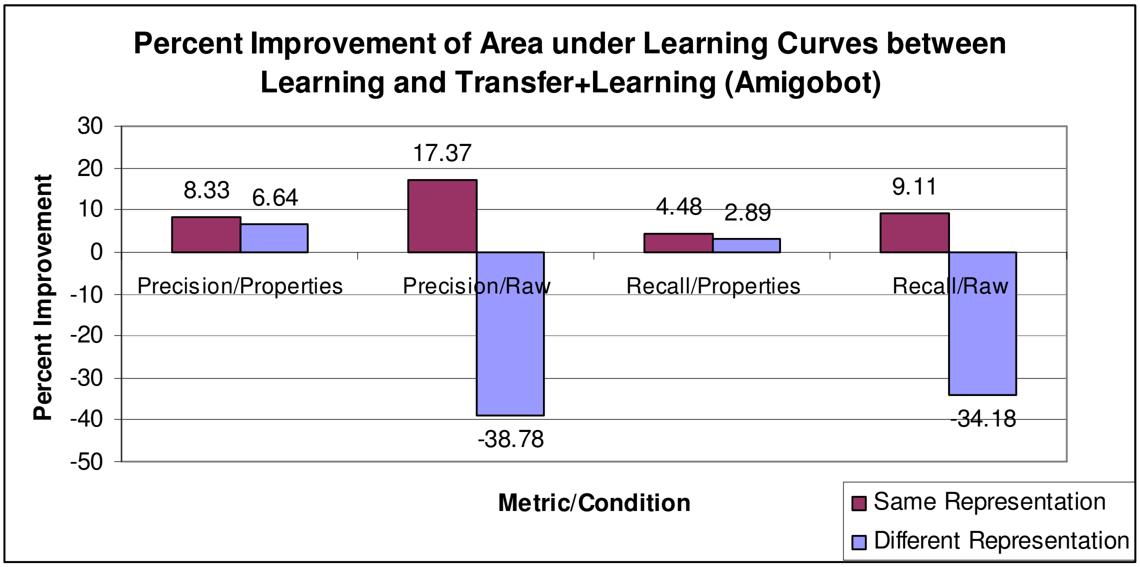


Figure 59 – Graph of percent improvement by transfer learning for Amigobot over self-learning in the area under the learning curves for both robots, recall and precision, and two conditions (same and different representations). When the robots used the same representation, transfer learning when using both property abstractions and raw sensory data yielded a net positive improvement. When using different representations, however, transfer learning with raw sensory data resulted in lower overall performance. This confirms our hypothesis that the property abstractions can aid knowledge transfer.

Transfer when using raw sensory data, however, did not result in the same gains and in some cases led to reduced overall effectiveness in learning. In order to quantitatively demonstrate the failure of transfer learning when using raw sensory data, we plotted the percentage difference in the area under the recall and precision learning curves when transfer learning is used relative to when self-learning is performed. Figure 59 shows the graph for the Amigobot robot. Positive values indicate an improvement over learning by one self while negative values indicate that transfer learning actually causes worse overall learning.

When property values are used, both recall and precision are improved over self-learning. This is true both when the same representation is used (8.33% precision improvement and 4.48% recall improvement) as well as when different representations are used by the two robots (6.64% precision improvement and 2.89% recall improvement). When using raw values, however, transfer learning is effective when the representations are the same (17.37% precision improvement and 9.11% recall improvement) but actually causes worse learning performance when different representations are used (38.78% *decrease* in precision rates and 34.10% *decrease* in recall rates). Even when the receiving robot continues to learn by itself (the "Transfer + Learning" curve), it fails to catch up to self-learning until a significant number of training instances are used. This is likely because the detrimental classifier received from the other robot must be overcome via training by the receiving robot. In this case, when using raw sensory values, it would likely have been better if the Amigobot had not received anything from the other robot. As shown quantitatively, transfer when using properties, however, remained as effective as before. This confirms the hypothesis that property abstractions can provide a buffer against underlying sensory differences and allows transfer learning to retain its advantages.

In this case, the results for the Pioneer robot differed. Figure 60 shows the same graph showing percent improvement via transfer learning over self learning. For the Pioneer, transfer did not cause a decrease in performance as for the Amigobot when using raw sensory data and different representations (there was an 8.22% precision improvement and 6.57% recall improvement). However, the transfer was still less effective than when properties were used (26.13% precision improvement and 15.12% recall improvement). Table 23 summarizes this experiment.

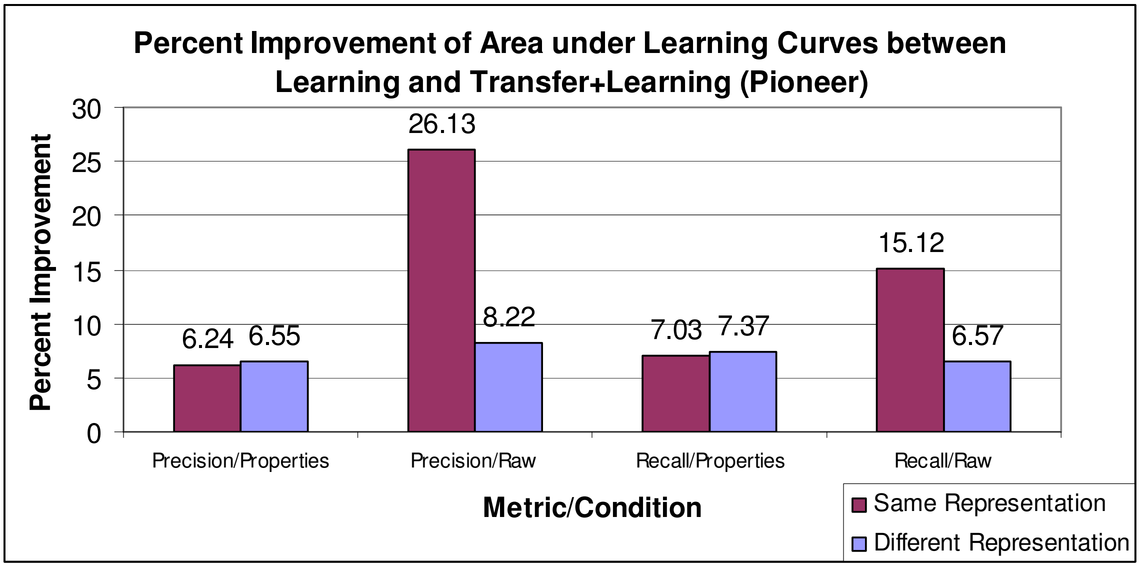


Figure 60 – Graph of percent improvement by transfer learning for Pioneer over self-learning in the area under the learning curves for both robots, recall and precision, and two conditions (same and different representations). When the robots used the same representation, transfer learning when using both property abstractions and raw sensory data yielded a net positive improvement. When using different representations, however, transfer learning with raw sensory data resulted in smaller gains.

5.6.4. Discussion

These results confirm our hypothesis that the abstraction of data into properties aids transfer, and is especially more effective than raw values when the robots utilize differing representations. We have now shown the value of abstracting raw sensory data both for learning (shown in experimental Section 3.10) *and* knowledge transfer (shown in this section). The abstraction of raw sensory data into higher level object properties such as color, texture, shape, and size serves as a buffer for lower-level differences in the robots. As a result, the knowledge that the robots learn becomes more transferable, as shown in the results.

Table 23 - - Experimental summary and conclusions for the experiment regarding the importance of property abstractions for knowledge transfer.

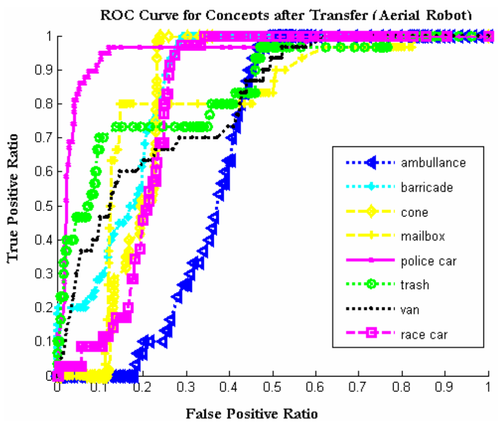
<table>
<tr><th colspan="2">Experiment 6: General Experiment Summary
The Importance of Property Abstractions for Transfer</th></tr>
<tr><td>Purpose</td><td>To determine whether the abstraction of raw sensory data into property abstractions aids transfer when robots use different underlying representations.</td></tr>
<tr><td>Experiment Type</td><td>Real-robot (configuration 2)</td></tr>
<tr><td>Hypothesis</td><td>We hypothesize that property abstractions do indeed aid transfer in the case of different representations. Specifically, we hypothesize that transfer learning will remain as effective when using properties but not when using raw sensory data.</td></tr>
<tr><td>Procedure</td><td>1. Train properties using labeled data.
2. Train two classifiers for concepts using labeled data
A. The input attributes to the classifier is raw sensory data
B. .The input attributes to the classifier is property memberships.
3. Train two classifiers for concepts using labeled data.
4. Measure improvement of transfer learning over self-learning by measuring the area under the learning curve for each condition.</td></tr>
<tr><td>Independent Variable</td><td>Input attributes to the classifier (raw sensory data vs. property memberships), Types of color space representations used (same representation vs. different representations).</td></tr>
<tr><td>Dependent Variable</td><td>Accuracy of concept classification as measured by recall, precision, and areas under the resulting learning curves as the number of training instances increases.</td></tr>
<tr><td>Conclusion</td><td>Hypothesis is confirmed. The improvement in terms of area under the learning curve decreased when using raw sensory data but not when using property abstractions.</td></tr>
</table>

Table 24 - Experimental summary and conclusions for the experiment demonstrating concept transfer for concepts represented within the conceptual spaces framework.

Experiment 7: General Experiment Summary Concept Transfer when using Conceptual Spaces	
Purpose	To determine whether robots can transfer concepts, represented in the conceptual spaces representation, classify concepts, and subsequently continue learning.
Experiment Type	Simulation, Real-robot (configuration 2)
Hypothesis	We hypothesize that the robots will indeed be able to transfer concepts between each other and classify concepts, and that the overall resulting performance throughout the robot's learning process will be better than without transfer (as measured by higher learning curves).
Procedure	1. Train properties using labeled data (algorithm in Table 5) 2. Train concepts using labeled data algorithm in Table 6) 3. Transfer concepts from one robot to the other (algorithm in Table 22) 4. Classify concepts in test data 5. Measure resulting performance
Independent Variable	Source of concept matrices used by the robot on test data (self-learned or transferred)
Dependent Variable	Accuracy of concept classification as measured by recall, precision, ROC, and areas under the resulting learning curves as the number of training instances increases.
Conclusion	Hypothesis is confirmed (Figures Figure 59 and Figure 60). There was improvement in terms of the area under the learning curves when transfer learning was used compared to self-learning.

5.7. Experimental Results: Concept Transfer when using Conceptual Spaces

Having established that abstracting raw sensory data via properties is important, we now show the same knowledge transfer capabilities using the conceptual spaces representation. Table 24 summarizes this experiment.



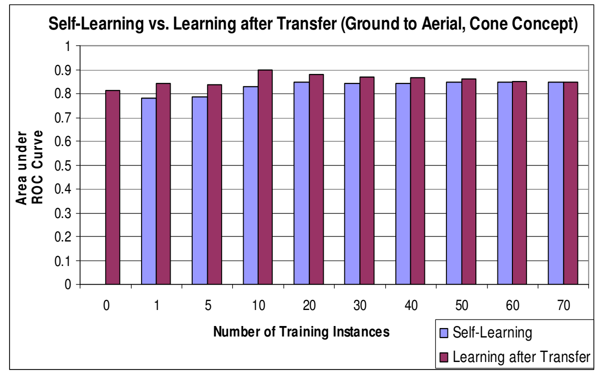


Figure 61 – ROC curve for concept categorization after concept transfer. The concept matrices used are from the other robot entirely. Right: Area under the ROC curve for concept categorization accuracy as new instances are combined with the representation received from the other robot (cone concept).

5.7.1. Hypothesis

We hypothesize that the robots will indeed be able to transfer concepts between each other and classify concepts, and that the overall resulting performance throughout the robot's learning process will be better than without transfer (as measured by higher learning curves).

5.7.2. Simulation Results

5.7.2.1. Procedure

We first demonstrate knowledge transfer results in simulation (Kira, 2009b). This experiment builds upon the first simulation experiment presented in Section 3.8.3. Recall that first, property representations were learned using methods presented in 3.5, and specifically the algorithm in Table 5. After the property representations were learned, there was a second training period during which the concepts (i.e. objects) themselves were learned. Concept learning was performed as described in Section 3.5 and specifically the algorithm in Table 6. Once learning was performed, we transferred the resulting concept matrices between the robots.

Given the transferred concept matrices, the receiving robot then classified concepts using the received representations. We also measured performance when the receiving robot used its own learned representations as a comparison. We measured performance using receiver operator curves (ROC). The ROC plots show the true positive ratio against the false positive ratio. The true positive ratio, or sensitivity, of a binary classifier is the number of true positives divided by the total number of positives in the test set. The false positive ratio is the number of false positives divided by the total number of negatives in the test set. The best possible classifiers would lie at the upper left corner, corresponding to only true positives and no false positives. A classifier performing at chance would lie on the diagonal line going from (0,0) to (1,1). One measure of total accuracy that can be used is the area under the ROC curve, where a value of one is perfect. A classifier performing at chance would result in an area of 0.5.

5.7.2.2. Results

We transferred all learned concepts from the ground robot to the aerial robot, and vice-versa, and tested the resulting accuracy. Learned property mappings that do not match, according to the property mappings learned, were used to modify the matrix as described previously in Section 5.4. Note that the concept matrix is transferred, but during categorization the receiving robot's own property memberships are used. Figure 61 shows the resulting ROC curves for the aerial robot. Most objects were categorized with very similar accuracy, despite the robot never having seen any instances of the concept.

Specifically, the mean area under the ROC curve for the ground robot was 0.77 and 0.81 for the aerial robot, only slightly worse than when the robot learns itself (0.82 for the ground robot and 0.89 for the aerial robot, or a difference of 6.1% and 9.0%).

Our last simulation result shows that the robots can take the concept representation given to it by the other robot, and continue learning using new instances. We do this by averaging the received concept matrix with the newly learned concept matrix. With this additional learning, the accuracy of the robots closely approaches the accuracy had the robot learned the concept itself from the beginning (3.7% difference for the aerial robot and 10.6% difference for the ground robot). Figure 61 shows the area under the ROC curve for one concept (cone), after an increasing number of new training instances are seen by the receiving robot. As can be seen, eventually the two performances almost converge (with a final difference of 0.15%). Note that for easier concepts where learning from a few instances results in high accuracy, the transferred concept does perform slightly worse than when the robot learns the concept itself. In all cases, though, the performance is comparable and if further training is performed the performances converge. This shows that a robot can receive a new concept from another robot (despite heterogeneity), successfully categorize the new objects almost as well as if the robot had learned the concept itself and then can continue learning. For harder concepts, such bootstrapping can allow the robot to perform well until it learns the concept itself.

Note that the improvements of knowledge transfer in simulation are not as great as for real robots, as will be shown in the next subsection. This is because learning in simulation was much easier, and in fact it only took a few instances to learn a concept well (e.g. five instances to get within 1.75% of the best measured performance for the aerial robot). This raises an interesting point regarding transfer learning. In previous results shown for real robots, we have shown that *as learning becomes harder the transfer of knowledge becomes even more advantageous and important* (Section 5.6, particularly Figure 59 and Figure 60). Conversely, the results here suggest that *if learning is extremely easy and only requires an instance or two, then the transfer of knowledge becomes less advantageous.*

5.7.3. Real-Robot Results (Configuration 2)

5.7.3.1. Procedure

This experiment builds upon the previous experiments with robots in this configuration (specifically, Section 3.8.4). One robot (the *expert robot*) first learns concept matrices for a set of concepts, builds similarity models with another robot based on the procedures described in the previous chapter (Section 4.2.1 and specifically algorithm in Table 16), and transfers the concepts using the algorithms outlined (Table 22). The non-expert robots differ from the expert robot in many ways, including sensor, feature, and property differences. The concepts are then tested on the receiving (non-expert) robot, using its own sensors but using the representations it received from the expert robot. After this first phase of testing, the receiving robot then continues to learn using labeled instances to show that it can continue learning successfully, adding to the representations received by another robot.

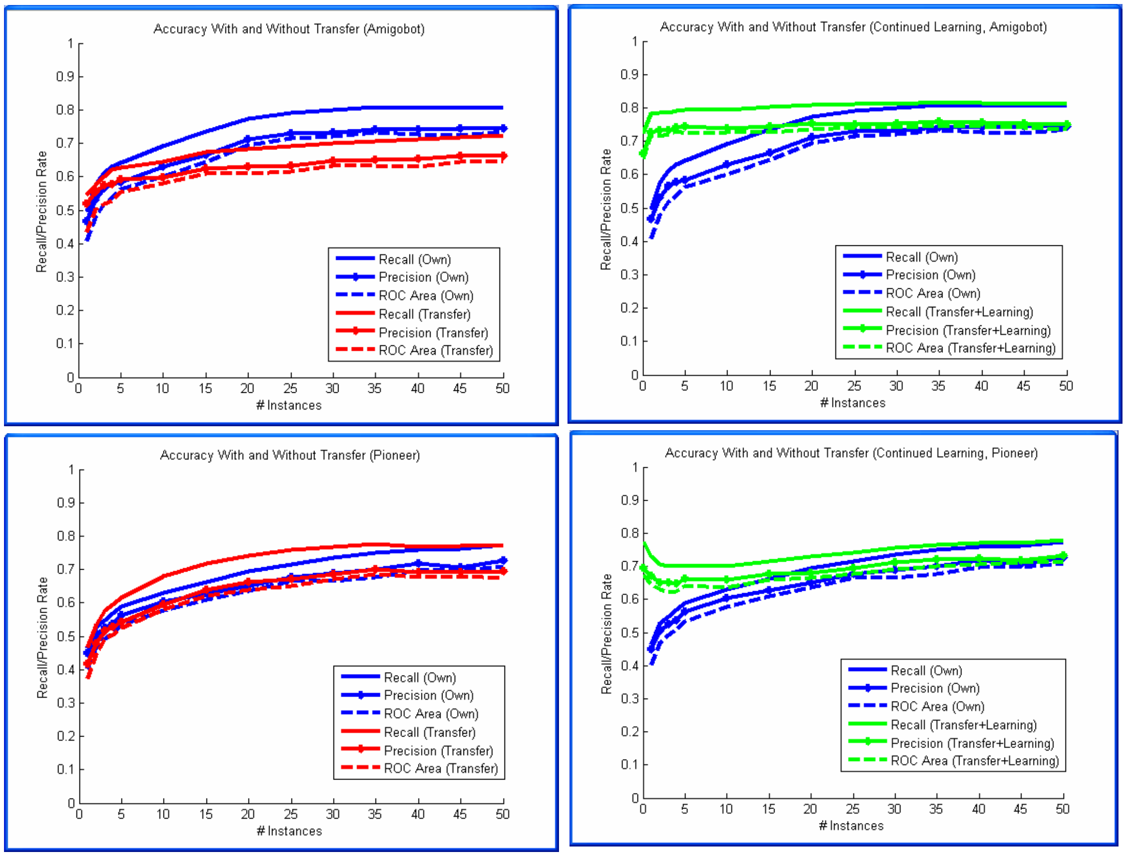


Figure 62 – Results demonstrating the success of transfer learning when using conceptual spaces, for both robots. In this case, both robots used the RGB color space for color properties.

5.7.3.2. Results

Figure 62 shows the results. The left portion shows "Own Training" versus "Transfer", as before. These graphs show the ROC area in addition to the recall and precision. As can be seen, transfer learning is successful here just as before. There is a significant advantage for the recall and precision rates achieved immediately after transfer without the robots having seen any instances when compared to the robot learning by itself for one instance (72.2% versus 49.5% for recall and 66.4% versus 46.6% for precision on the Amigobot, and 77.3% versus 46.3% for recall and 69.4% versus 45.0% for precision on the Pioneer).

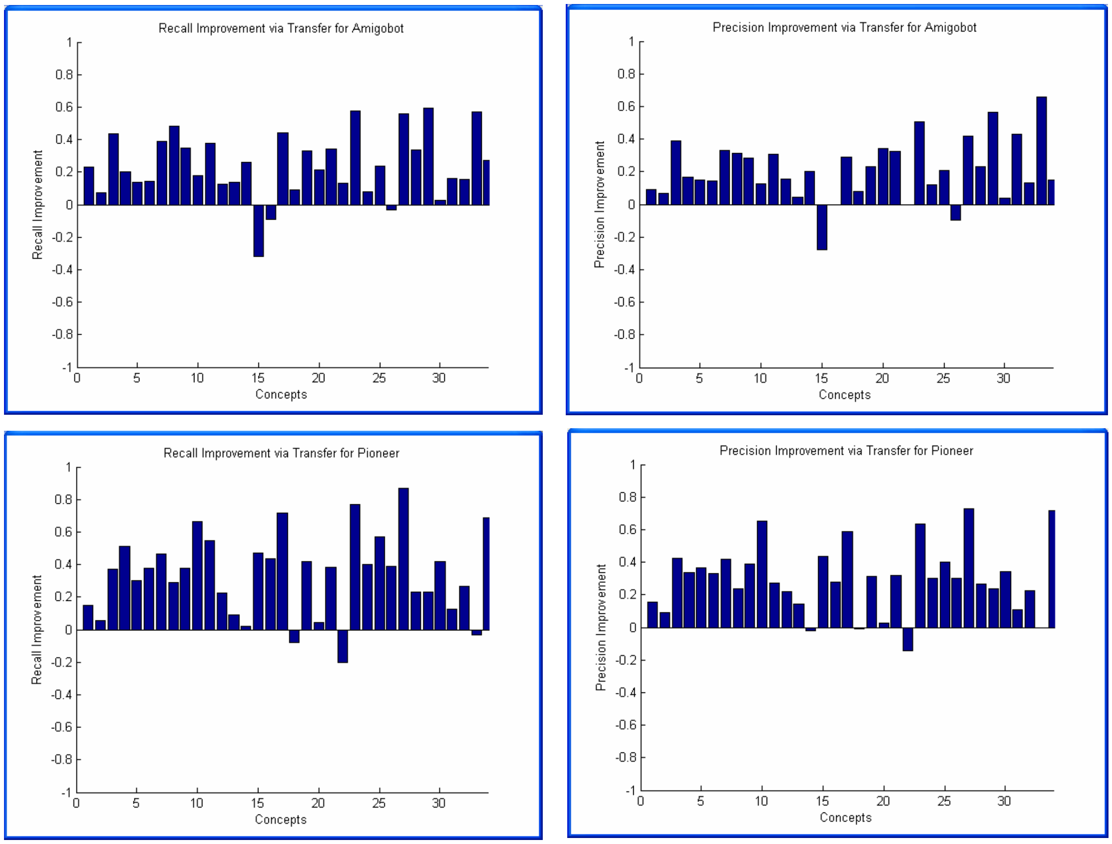


Figure 63 – Bar graph showing recall (left) and precision (right) improvements when using knowledge transfer compared to learning after only one instance.

Interestingly, the Pioneer robot (which has additional properties that the Amigobot does not) goes through a period of adaptation for approximately fifteen instances in the beginning after transfer and continued learning (green curves on lower right panel of Figure 62). The transferred representation achieves high accuracy (69.4% precision, 77.3% recall), after which it dips (to 66% precision and 70.0% recall) and then begins to move upward again (finishing at 73.1% precision and 77.9% recall). We conjecture that this is likely because the concept matrix at first does not include the properties that it does not share with the Amigobot. After it begins to learn using its own instances, these begin to be filled in. As the number of such instances increases, the Pioneer robot then begins

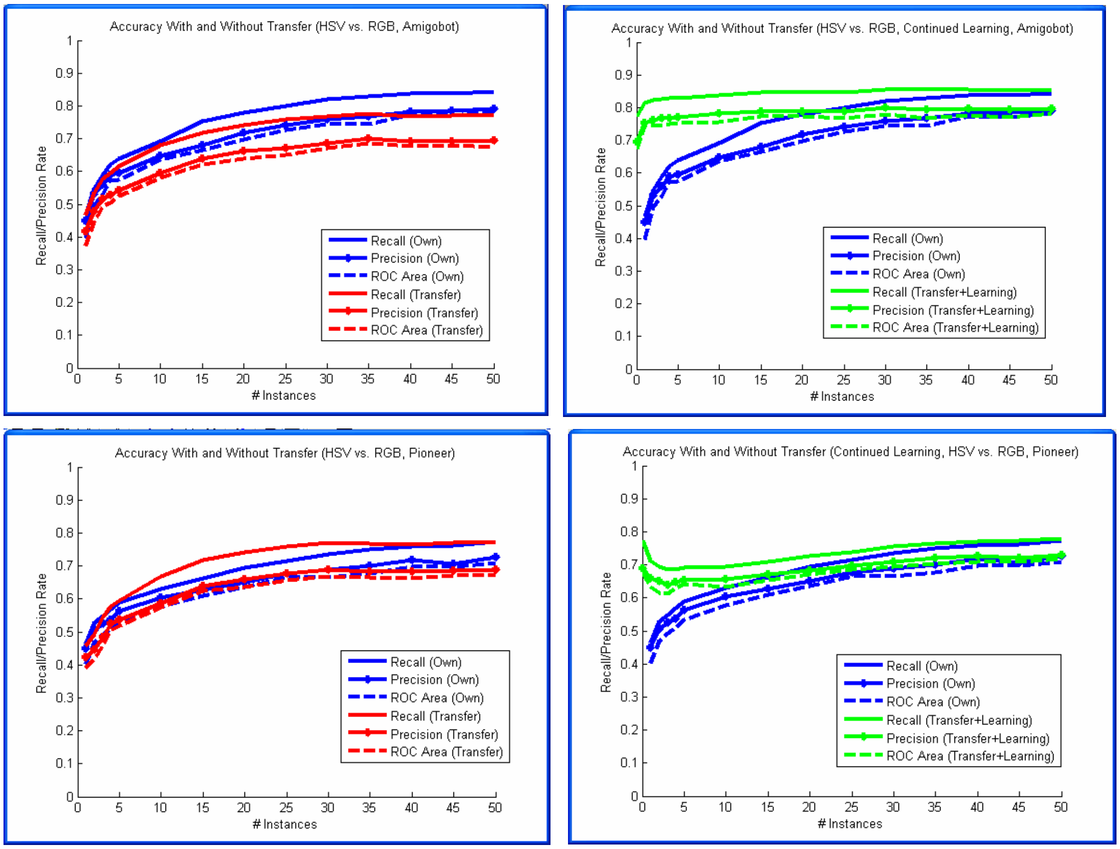


Figure 64 – Results demonstrating the success of transfer learning when using conceptual spaces, for both robots. In this case, the Amigobot used an HSV color space for the color properties while the Pioneer used an RGB color space. Again, transfer learning is advantageous, as seen by higher initial performance (0 instances) as well as higher learning curves for "Transfer+Learning" condition over "Own" condition.

to move back towards its maximum performance. Figure 63 shows the bar graphs that show the recall and precision improvements for each concept, compared with only one instance when the receiving robot learns by itself. Again, a majority of concepts (all but three for recall, or 91%, and all but two for precision, or 94%) obtain an advantage from knowledge transfer. Figure 64 shows the learning curves with and without transfer when the two robots use different underlying spaces for the color properties (HSV versus RGB). Again, transfer learning is still successful compared to self-learning after one instance (77.3% versus 46.7% for recall and 69.4% versus 44.9% for precision on the Amigobot, and 77.5% versus 46.3% for recall and 69.0% versus 45.0% for precision on the Pioneer). Table 25 shows all of the relevant data including the areas under the curves and performance after the first, fifth, and final training instances.

Table 25 - Complete results showing the advantage of transfer learning over self-learning.

	Pioneer (Same Representations)								
	Self-Learning			**Transfer**			**Transfer + Learning**		
	Precision	**Recall**	**ROC Area**	**Precision**	**Recall**	**ROC Area**	**Precision**	**Recall**	**ROC Area**
Area	0.64	0.68	0.62	0.64	0.71	0.62	0.67	0.72	0.66
First	45.01	46.29	40.17	41.80	46.62	37.38	69.40	77.28	67.47
Fifth	56.23	58.78	53.42	54.12	61.47	52.55	64.85	70.07	62.39
Final	72.67	77.23	70.93	69.40	77.28	67.47	73.10	77.88	71.42
	Amigobot (Same Representations)								
	Self-Learning			**Transfer**			**Transfer + Learning**		
	Precision	**Recall**	**ROC Area**	**Precision**	**Recall**	**ROC Area**	**Precision**	**Recall**	**ROC Area**
Area	0.68	0.74	0.66	0.62	0.67	0.60	0.73	0.79	0.72
First	46.59	49.48	40.88	51.84	54.48	43.48	66.35	72.23	64.76
Fifth	58.25	64.23	56.31	59.24	62.59	55.41	74.18	79.00	73.10
Final	74.38	80.62	73.42	66.35	72.23	64.76	74.95	81.02	73.59
	58.25	66.35							
	Pioneer (Different Representations)								
	Self-Learning			**Transfer**			**Transfer + Learning**		
	Precision	**Recall**	**ROC Area**	**Precision**	**Recall**	**ROC Area**	**Precision**	**Recall**	**ROC Area**
Area	0.64	0.68	0.62	0.63	0.71	0.61	0.67	0.71	0.66
First	45.01	46.29	40.17	42.18	45.67	39.04	68.95	77.48	67.45
Fifth	56.23	58.78	53.42	53.74	59.58	51.92	64.72	68.62	63.10
Final	72.67	77.23	70.93	68.95	77.48	67.45	73.03	77.80	71.69
	Amigobot (Different Representations)								
	Self-Learning			**Transfer**			**Transfer + Learning**		
	Precision	**Recall**	**ROC Area**	**Precision**	**Recall**	**ROC Area**	**Precision**	**Recall**	**ROC Area**
Area	0.70	0.76	0.69	0.64	0.71	0.62	0.77	0.83	0.75
First	44.95	46.71	39.72	41.80	46.62	37.38	69.40	77.28	67.47
Fifth	59.62	63.89	57.49	54.12	61.47	52.55	76.80	82.78	74.86
Final	79.15	84.38	78.17	69.40	77.28	67.47	79.68	85.30	78.15

5.8. Experimental Results: Estimating Post-Transfer Performance

This chapter's final real-robot learning experiment investigates the ability of the expert robot to estimate, using its own performance and only *shared* properties, how effectively the receiving robot will perform with the transferred representations.

5.8.1. Hypothesis

We hypothesize that the sending robot will be able to estimate the performance of the receiving robot when classifying test instances using transferred concepts.

5.8.2. Procedure

Table 26 - Conditions used for experiment, where random subsets of properties were used (the first condition used all properties).

Condition #	Properties Used
1	1 2 3 4 5 6 13 14 15 16
2	1 2 3 4 5 13 14 15
3	1 2 3 4 6 14 15 16
4	1 2 4 13 14 16
5	1 13
6	1 4 5 13
7	2 3 16
8	2 13
9	14
10	1 2 3 4 5 6

We used nine random configurations, representation random *subsets* of the actual properties that they share. In addition, we also used one fixed configuration that uses all actually shared properties. Table 26 shows the configurations. In this case, the Amigobot was the expert (transferring) robot while the Pioneer was the non-expert

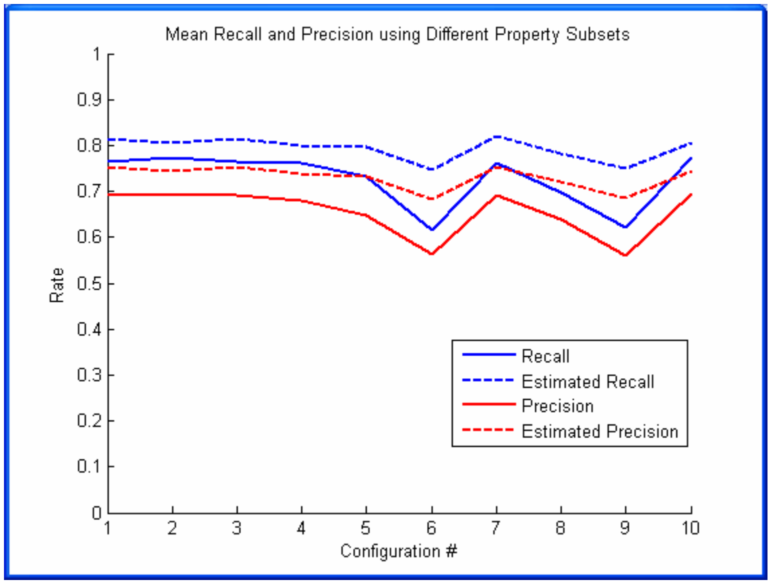


Figure 65 – Plot of the estimated and real recall and precision. The Amigobot was the transferring robot, and hence its performance was used as an estimate (red). The estimates were accurate with a mean difference of only 0.072 for recall rate and 0.033 for precision rate.

(receiving) robot. For each configuration, the Amigobot used the randomly chosen subset of properties to classify all concepts. These recall and precision results were used as estimates in terms of how accurate the classification will be once these concepts are transferred to the Pioneer. For each configuration, we compared the difference between the estimate and the actual resulting accuracies. Two comparisons were made. First, we compared the *mean*s of the estimated accuracies over all concepts. The estimates and real performance values are shown in Figure 65. We also compared the estimated accuracies concept by concept, and hence measured the mean absolute difference between the estimation and occurring accuracies. In other words, we measured the difference between the means and the mean of the absolute differences. We measured both of these in two conditions, "Own" and "Transfer". One measured the differences between the actual performances in the two robots when both learned using their own respective instances ("Own"). The second condition measured the differences between the actual performance of the transferring robot (which represents the estimate) and the performance on the receiving robot using the transferred representations ("Transfer").

Table 27 - Results for transfer estimation. The difference in means had significantly smaller mean differences between estimate and real performance, while the mean difference had much larger means and standard deviation.

	Difference in Means		Mean Difference		
	Mean (N=10)	Std	Mean (N=10)	Std	P value
Recall	0.072	0.038	0.214	0.179	0.024
Precision	0.033	0.028	0.174	0.200	0.041

5.8.3. Results

Figure 66 shows all of the results and Table 27 shows the values for the "Transfer" condition along with significance tests. The panel on the left in the figure shows the difference in *means* between the estimated accuracy and actually occurring accuracy. The results are averaged over the ten configurations. As can be seen, the Amigobot can successfully use its mean performance (over all concepts) to estimate the mean performance of the Pioneer. The "Own" condition shows that the two robots achieve approximately the same mean accuracy over all concepts given the same subset of properties. The "Transfer" condition is not much different, showing that the mean performance achieved by the transferring robot can be accurately used as an estimate (within a recall rate of 0.07 and precision rate of 0.03) of how well the receiving robot will perform (on average over all concepts) given the transferred representation.

The right panel of Figure 66 shows the difference between estimated accuracy and actual accuracy taken on a concept-by-concept basis. The resulting errors in estimation are far larger with larger standard deviations than the previous comparison (see Table 27).

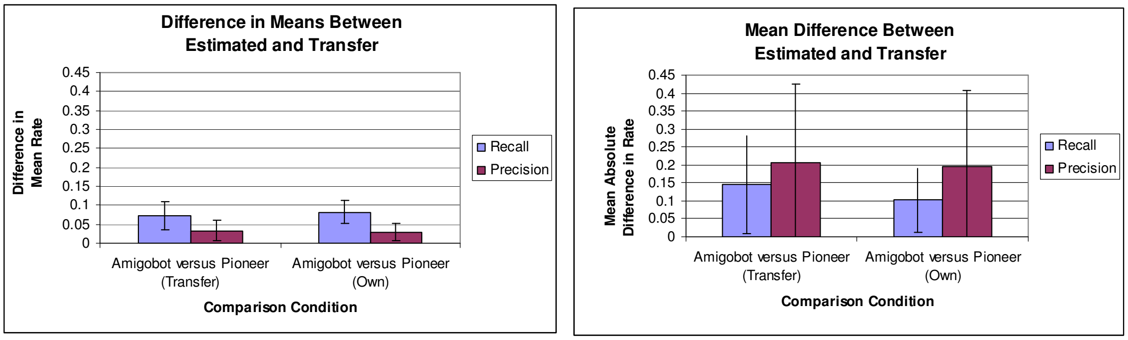


Figure 66 – Graphs demonstrating the efficacy of *a priori* estimation of the accuracy of the transferred representation. The left graph shows the effectiveness of estimation in terms of *mean* accuracy over all objects. The right graph shows the effectiveness of estimating the accuracy of *individual* concepts. As can be seen, it is more difficult to estimate the success of transfer for a particular concept, but the mean accuracy over many concepts can be estimated. The difference between the transfer estimates in the difference in means (a difference of 0.07 for recall rate and 0.03 precision rate) is significantly different than the mean difference (0.21 for recall rate and 0.17 for precision rate). The p values were 0.0242 for the recall rate and 0.0405 for the precision rate, representing significant differences (< 0.05). Both graphs show two conditions. The "Transfer" condition compares accuracy of estimation for transferred concepts, while the "Own" condition compares accuracy of estimation for concept accuracy when each robot learns on its own.

The difference between the transfer estimates in the difference in means (a difference of 0.07 for recall rate and 0.03 precision rate) is significantly different than the mean difference (0.21 for recall rate and 0.17 for precision rate). The p values were 0.0242 for the recall rate and 0.0405 for the precision rate, representing significant differences ($p<0.05$). Table 28 summarizes this experiment.

In other words, predicting how well transferring a particular concept will work results in much more uncertainty. If the transferring robot uses its own performance as an estimate, it will be off by an average rate of about 0.21 for recall and 0.17 for precision, with a large standard deviation in the estimation found in this case. This may or may not be acceptable depending on the task. In summary, the two robots perform about the same when averaged over all concepts (with a difference in the rates less than 0.1 for both recall and precision). In other words, the hypothesis that performance can be predicted on an average basis is confirmed. This can be used as a useful estimate of how well the

Table 28 - Experimental summary and conclusions for the experiment regarding the *a priori* estimation of transfer performance.

Experiment 8: General Experiment Summary **Estimating Post-Transfer Performance**	
Purpose	To determine whether the sending robot can estimate the performance of the receiving robot given knowledge of shared and unshared properties..
Experiment Type	Real-robot (configuration 2)
Hypothesis	We hypothesize that the sending robot will be able to estimate the performance of the receiving robot when classifying test instances using transferred concepts.
Procedure	1. Train properties using labeled data for both robots(algorithm in Table 5) 2. Train concepts using labeled data for both robots (algorithm in Table 6) 3. Transfer concepts from one robot to the other (algorithm in Table 22) 5. Estimate performance of receiving robot by the sending robot by converting the concept matrix to use only shared properties. 4. Compare estimates to actual receiving robot's Performance 5. Perform 1-4 for different subsets of shared properties
Performance Metrics	We calculate the difference between estimated performance (by the sending robot) and actual performance (by the receiving robot) both for estimation of performance of a single concept as well as estimation of mean performance on all concepts.
Conclusion	Hypothesis is not confirmed for estimation of individual concepts. Predicting performance on a per-concept basis resulted in higher means and variances than predicting performance of mean performance over all concepts. Hypothesis that performance can be predicted on an average basis is confirmed.

receiving robot will perform given a particular set of shared properties. This capability, by itself, can be potentially useful. For example, suppose two robots are performing a particular task that requires the recognition of a set of concepts. The expert robot can determine, given shared properties, whether the receiving robot will be able to perform that task well. However, it is not guaranteed that the two robots will perform the same on any given concept. Hence, the hypothesis is not confirmed for estimation of individual concepts. Predicting performance on a per-concept basis resulted in higher means and variances than predicting performance of mean performance over all concepts. It is much more problematic due to performance differences between the robots to use the performance of the transferring robot on a particular concept to estimate the performance of the receiving robot on that same concept.

5.9. Summary

This chapter has laid a foundation for the transfer of knowledge across heterogeneous robots. We began by exploring how robots can differ at different levels of representation, and proposed an interactive process that determines how to align these differences. The most useful communication can occur when the robots have some overlap in the properties they use to represent concepts. Given such overlapping properties, which are found via algorithms detailed in Chapter 4, two robots can transfer concepts between each other. We showed how concepts, represented as matrices as described in Chapter 3, can be *adapted* to the receiving robot before it is transferred.

After detailing the framework and algorithms, we conducted several experiments. First, in Section 5.6, we have provided evidence for our hypothesis that abstracting raw sensory data into properties aids in transfer, for example by making it possible despite differences in the underlying property representation used by the robots (Kira, 2010). Properties effectively provide a buffer against differences in the lower-level sensors and features. We then demonstrated, using real robot data, that robots can indeed effectively transfer concepts given knowledge of what properties are shared between them. This was shown using both support vector machines (Section 5.6) and conceptual spaces (Section 5.7), demonstrating generality across different concept representations. We also showed quantitatively that the advantages of knowledge transfer occur for almost all of the thirty-four concepts we have tried (Figure 56 and Figure 63).

Further experiments were also performed to test whether the classification accuracy of the expert robot, using only the shared properties, can be used to estimate how effectively the receiving robot will perform (Section 5.8). We showed that such estimations are problematic on a concept-by-concept basis due to large inaccuracies and standard deviations. However, *average* performance over a number of concepts can be accurately estimated, as shown.

Finally, we have shown that the same principles can work in simulation (Section 5.7.2) (Kira, 2009b). However, since learning is easier due to lack of significant noise, knowledge transfer is less advantageous. Despite this, transfer still allows the receiving robot to classify new concepts with no training.

Now that we have established a successful framework for knowledge transfer, the remaining chapters will explore the resulting capabilities further. Until now, we have assumed that two robots either share a property or not, i.e. the mapping is binary. One of the advantages of using supervised learning for properties was that we could control the learning such that both robots learned similar properties. We did this by using the same objects with consistently similar properties as training. In the next chapter, we will look at how to handle knowledge transfer in the situation where properties between robots only overlap partially, by varying the property training regime in the two robots.

CHAPTER 6

MEASURING INFORMATION LOSS IN CONCEPT TRANSFER

In the preceding chapters, we have described methods for transferring learned knowledge between robots given a mapping between their respective properties represented as confusion matrices (Section 4.2.1). Membership values for those properties that were shared were kept in the transferred concept matrices, while those that were not shared resulted in adaptation of the learned knowledge (Section 5.4). In other words, the mapping between properties was completely binary: shared or unshared. In this chapter we look at the case where properties may overlap, and we measure the amount of overlap using an information-theoretic metric. We show that this metric correlates with the performance of transfer learning. In Chapter 7, we will leverage the entire framework, including this metric, to perform several types of communication tasks such as picking the most similar robot.

6.1. Property Overlap: A Source of Error in Concept Transfer

In Section 5.3 we discussed potential sources of error during concept transfer between two robots. Recall that the purpose of analyzing sources of error was to be able to estimate *a priori* the performance of knowledge transfer. This is important for cognizant failure; that is, the ability to know when the transfer of knowledge will result in unacceptable performance. Note that in general the methods presented in Section 5.3 and its associated experiments (Section 5.8), as well as the methods below, only produce *estimates* of post-transfer performance. There can be additional sources of error such as

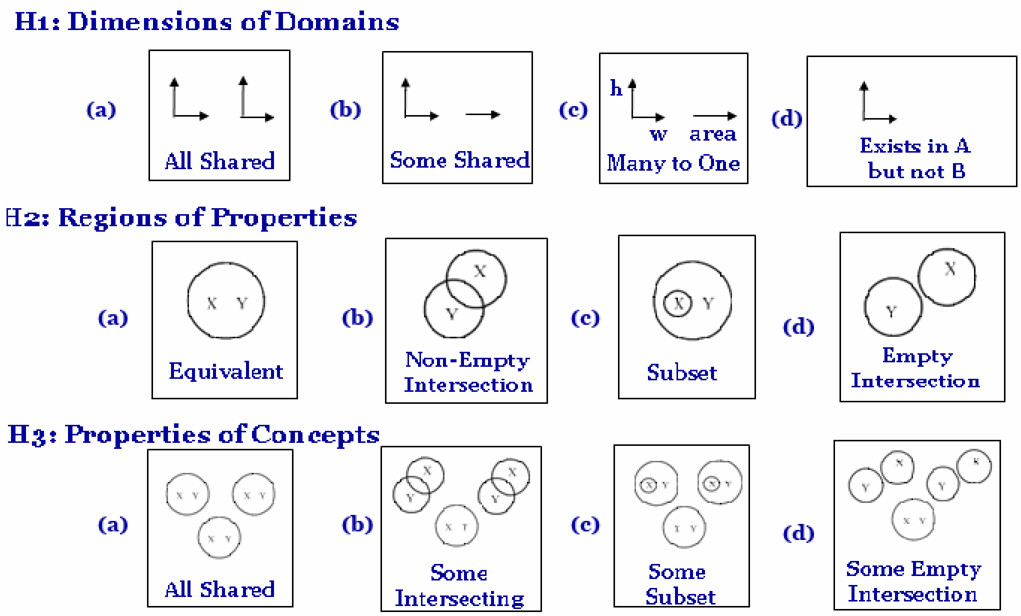


Figure 67 – Classes of heterogeneity defined by differences in multiple levels of representation. In Chapter 4, we looked at properties as either shared or unshared (Types H2a, H2d, H3a, and H3d). This chapter deals with types 2b and 3b, where properties may overlap partially.

random noise (for example, in the property memberships), poor recall rates of the properties, effects due to combinations of properties working in concert, etc.

In Section 5.3, we identified two major sources of error: 1) Unshared properties and 2) properties that only partially overlap. In some sense, the first source of error is really an edge case of the second where the degree of overlap between properties is zero. In other words, unshared properties are properties that are completely non-overlapping. Properties that partially overlap will have some correlation and therefore have non-zero values in the confusion matrices learned in 4.2.1 (algorithm in Table 16). For each property on one robot, the property mapping algorithm will select the property on robot two that has the highest overlap. However, we have thus far treated mapped properties as equivalent and have not used the amount of overlap in order to estimate the performance of concept transfer. In terms of the heterogeneity types defined in Section 3.6 (see Figure

67 for a summary), we considered properties as shared or unshared (heterogeneity types H2a, H2d, H3a, and H3d) (see Chapter 4). In this chapter, we will consider the degree of overlap with potential heterogeneity H2b and H3b.

6.1.1. Defining Property Overlap

Heterogeneity types H2b and H3b specify that properties can partially overlap. The notion of property overlap has a physical meaning when properties are represented as Gaussians. Specifically, one can measure the physical overlap of the hyper-ellipsoids. However, the Gaussians representing properties do not necessarily exist in the same space for both robots. Similar to the discussion in Section 4.2, instead of comparing the property representations directly, an interactive process is used instead. The robots achieve a shared context (Section 4.4), calculate property memberships, and use these memberships to build a model of the amount of overlap. The previous method used fuzzy confusion matrices as the model (Section 4.2.1). Note that we use the term *fuzzy* due to the usage of the *min* function, not in the sense of standard fuzzy logic. In this section, we use a metric grounded in information theory to actually measure the amount of information loss when moving from one property representation to another.

6.2. Variation of Information Metric

In order to define and calculate information loss, we again take inspiration from methods used to compare clusterings in machine learning. Specifically, we use the variation of information metric (VI, described below) used to compare clusterings (Meila, 2002). In our case, we consider each property pair (one from each robot) and calculate the variation of information between them. The clusters correspond to the property in robot 1, all other parts of the space not in the property for robot 1, the property in robot 2, and all other parts of the space not in the property for robot 2. In other words, for each robot we consider the property to be used as one cluster and the rest of the space as the other cluster. Hence, there are four clusters in total.

We begin by defining a discrete random variable for each cluster (in our case a property), g, representing the probability that an instance (picked out at random) will belong to that cluster (Meila, 2000). Assuming each instance has an equal probability of being picked, and that all instances belong to a cluster, it can be expressed as:

$$P(g) = \frac{n_g}{n} \qquad \textbf{(9)}$$

where n_g is the number of instances in cluster g. Since in our case the membership to a property (or cluster) is continuous, this value can instead be calculated by taking the mean property membership across all instances. To calculate the probability of belonging to the second cluster (i.e. not the property), we subtract the probability of being in the property from one (1.0). The entropy of this random variable is expressed as:

$$H(G) = -\sum_{g=1}^{n_i^A} P(g) \log P(g) \qquad \textbf{(10)}$$

This defines the entropy associated with one clustering G. We can now define *mutual information* between two clusterings. Let $P(g, g')$, where $g \in G_i^A$ and $g' \in G_j^B$, be the probability that an instance belongs to both clusters, i.e.

$$P(g, g') = \frac{|g \cap g'|}{n} \qquad \textbf{(11)}$$

This probability can be estimated using the instances. The mutual information between the two clusterings can finally be defined as:

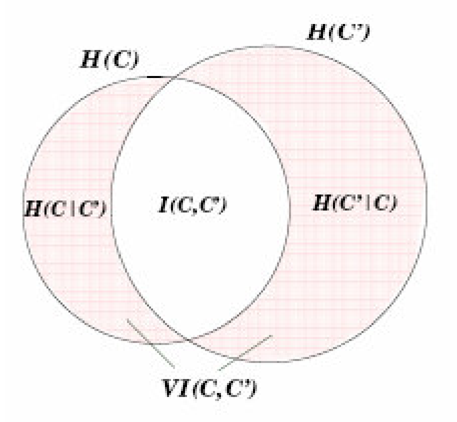


Figure 68 – Venn diagram depicting relationship between entropy, conditional entropy, variation of information, and mutual information for two clusterings (Meila, 2000).

$$I(G_i^A, G_j^B) = \sum_{g=1}^{n_i^A} \sum_{g'=1}^{n_j^B} P(g, g') \log \frac{P(g, g')}{P(g)P(g')} \qquad \textbf{(12)}$$

The relationship between the entropy of each clustering, conditional entropies, and mutual information can be seen in Figure 68. The variation of information (VI) is then defined as:

$$VI(G_i^A, G_j^B) = H(G_i^A) + H(G_j^B) - 2I(G_i^A, G_j^B) \qquad \textbf{(13)}$$

or alternatively:

$$VI(G_i^A, G_j^B) = [H(G_i^A) - I(G_i^A, G_j^B)] + [H(G_j^B) - I(G_i^A, G_j^B)] \qquad \textbf{(14)}$$

This metric measures the *loss of information* when going from one clustering to another. Intuitively, this is what we want to measure when communication occurs between two robots. If the information loss is greater than a certain empirically-determined threshold, then we consider the properties to be not shared; i.e., p_i^A and p_j^B are considered shared if and only if $VI(G_i^A, G_j^B) < \theta$. Note that the threshold depends largely on the task. If a high classification rate is extremely important, for example in a search and rescue task where a life may depend on it, then a fairly tight threshold is needed. If the task is entertainment by a household robot, then a looser threshold can be employed. If the property pairs do not meet the criteria, the pair can be removed from the property mapping model. We also use this measure of information loss when sharing entire concepts, or between entire sets of properties that robots have, which can be defined by combining the information loss of its constituent properties. In Chapter 7, we will utilize this to perform such tasks as picking the most similar robot from a conceptual standpoint in order to improve knowledge transfer.

6.3. Calculating the Variation of Information from Data

Table 29 details the algorithm implementing the equations above and shows how to calculate the variation of information metric using instances. First, instances from a shared context (Section 4.4) are gathered. Property memberships are then calculated for each robot. These membership values are then used to calculate the metric, as described above as well as in the algorithm below.

Table 29 – Algorithm for calculating the variation of information metric from data

Algorithm: Calculating the Variation of Information Metric from Data

```
Input: Properties P1, P2 (from robot 1 & 2)
Output: Variation of Information Matrix VI

// Establish a shared context.  This can be done using manually-picked images
// viewing the same scene, or robot behaviors such as following, pointing, etc., e.g.
// as described in (Kira & Long, 2007).

For each shared context instance si

    // Calculate property memberships for both robots
    Values1 = P1(si)
    Values2 = P2(si)

    // Add them to training instances
    Instances.add(si, Values1, Values2)
```

```
End

// Calculate VI for each property pair
For each property p1 in P1
  For each property p1 in P2

     // Get property memberships for the two properties for all instances
     Values1 = gi.Values1(p1) across all instances
     Values2 = gi.Values1(p2) across all instances

     // Calculate P(g) and P(g')
     P_g = Mean(Values1)
     P_gp = Mean(Values2)

     // Calculate the entropy of the random variable (Equation 10)
     H_G = -(P_g .* log2(P_g)) - ((1-P_g) .* log2((1-P_g)));
     H_Gp = -(P_gp .* log2(P_gp)) - ((1-P_gp) .* log2((1-P_gp)));

     // Calculate I_G_Gp (Equation 12)
     I_G_Gp = 0;

        this_cluster = [];
        that_cluster = [];

        // Here, we consider the four clusters (p1, not p1, p2, and not p2)
        // We calculate the individual probabilities, where the probability of not
belonging to
        // a cluster is one minus the probability of belonging to the cluster, since there
are
        // only two clusters spanning the entire space.
        for k=1:4
           if (k == 1)
              // Situation 1: Instances from robot 1 that belong to property p1 and
instances
              // from robot2 that belong to property p2 as well
              this_P_g = P_g;
              this_cluster = Values1;
              that_P_gp = P_g;
              that_cluster = Values2;
           else if (k == 2)
              // Situation 2: Instances from robot 1 that do not belong to property p1 and
              // instances  from robot2 that belong to property p2
              this_P_g = 1-P_g;
              this_cluster = 1-Values1;
              that_P_gp = P_g;
              that_cluster = Values2;
```

```
          elseif (k == 3)
             // Situation 3: Instances from robot 1 that belong to property p1 and
             // instances  from robot2 that do not belong to property p2
             this_P_g = P_g;
             this_cluster = Values1;
             that_P_gp = 1-P_g;
             that_cluster = 1-Values2;
          elseif (k == 4)
             // Situation 4: Instances from robot 1 that do not belong to property p1 and
             // instances  from robot2 that do not belong to property p2
             this_P_g = 1-P_g;
             this_cluster = 1-Values1;
             that_P_gp = 1-P_g;
             that_cluster = 1-Values2;
          end

          // Calculate joint probability (represented as a fuzzy logic operation, the min
          // function)
          P_g_gp = mean( min(Values1, Values2) );

          // Mutual information between two clusterings
          if (P_g_gp > 0)
             I_G_Gp = I_G_Gp + (P_g_gp * log2(P_g_gp / (this_P_g*this_P_gp)));
          end
       end

       // Calculate variation of information metric
      VI(p1,p2) = H_G + H_Gp - 2*I_G_Gp (Equation 13)
  End
End

Return VI
```

6.4. Experimental Results: Calculating the Variation of Information Metric

This first experiment seeks to determine whether the variation of information metric correlates with the degree of overlap in properties. We seek to determine whether this is the case when the metric is calculated using real test data obtained from a manually-guaranteed shared context (as in Section 4.5.3). Here, the fact that properties are human-supervised becomes very useful as the degree of overlap can be controlled experimentally. The method for controlling this variable is described in Section 6.4.2.

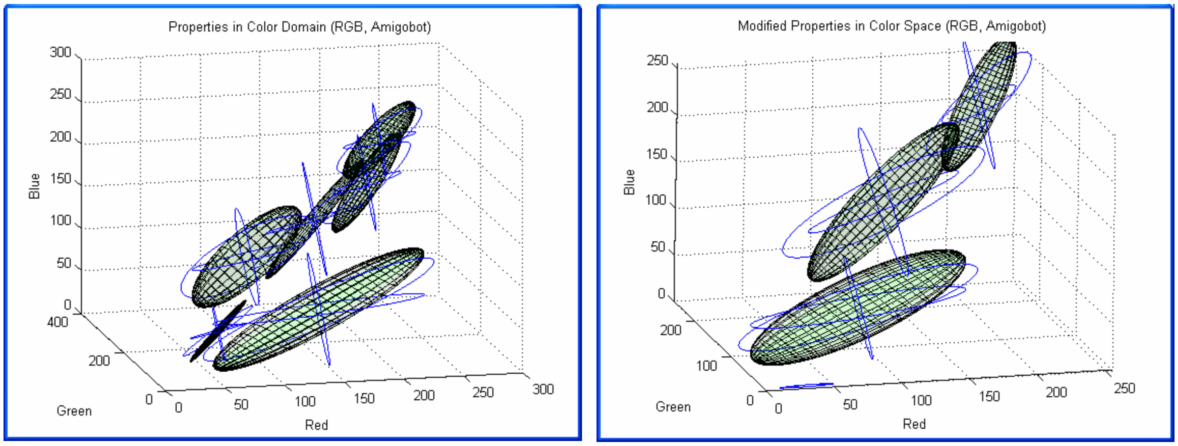


Figure 69 – This figure shows how the properties on the Amigobot were modified in order to change their overlap with the properties on the Pioneer. Left: The original properties. Right: The modified properties. The modified properties represent two properties merged into one. Since the other robot's representation was not modified, there will be less overlap between the two respective property representations. The blue ellipses represent the covariances.

6.4.1. Hypothesis

The hypothesis of the experiment is that as the amount of overlap between the properties is varied (as controlled experimentally), the variation of information metric will linearly increase. In other words, we hypothesize that the metric, when calculated using real robot data, reflects the degree of overlap between properties.

6.4.2. Procedure

In order to control the amount of overlap between properties, the training regime used by the Amigobot in prior experiments (Section 3.8.4) was modified. Specifically, the training instances for two color properties were used to train only one property. This resulted in the merging of two property representations. This was done for three pairs of the color properties (there were six color properties in total). The resulting properties can be seen in the right side of Figure 69, and are much larger than the original properties (left side of Figure 69). This merged property representation only partially overlaps with the corresponding properties on the other robot since the latter ones were not modified.

Figure 70 – An example image pair used to calculate the variation of information metric between the two robots. Left: Image from the Amigobot. Right: Image of the same scene from the Pioneer robot.

The properties were modified for the Amigobot robot, and the variation of information metric was then calculated as described in Section 6.3 (algorithm in Table 29) and experimental Section 4.5.3. The metric was averaged over all shared property pairs. Figure 70 shows an example image pair used to calculate the metric. This first experiment serves to show whether using these properties as opposed to the original properties will result in increases of the variation of information metric. This shows that the metric does indeed vary with different degrees of overlap. The next experiment will analyze the effect of this modification on knowledge transfer.

Eight different conditions were used for this experiment. The eight possibilities correspond to a decision of whether to replace a pair of properties with the merged representation or whether the original properties (that are split) should be used. Since there are three pairs in this case and a binary decision for each, this resulted in eight combinations. Table 30 shows these possibilities. The hypothesis is that the variation of information metric will be least when all of the original properties are used (since they were learned using the same training regime as the other robot), while the last combination where all pairs are replaced with the new regime should yield the highest variation of information.

Table 30 – Eight conditions used for the first experiment. A "No" means that the property pair was not merged (i.e. the original representation for the two properties was used). A "Yes" means that the instances for the property pair was treated as one and a merged representation was created.

	Merge Representation for:		
Condition	**Properties 1 & 2**	**Properties 3 & 6**	**Properties 4 & 5**
1	No	No	No
2	No	No	Yes
3	No	Yes	No
4	No	Yes	Yes
5	Yes	No	No
6	Yes	No	Yes
7	Yes	Yes	No
8	Yes	Yes	Yes

Table 31 – Variation of Information (VI) results for each condition. The column displaying the number of properties with partial overlap counts the number of "Yes" entries for that row. For each "Yes", a merged property representation is used instead of the original properties. This merged property representation will only partially overlap with the corresponding separate properties on the other robot. As the number of properties with partial overlap increased, so did the VI metric.

	Merge Representation for:				
Condition	**Properties 1 & 2**	**Properties 3 & 6**	**Properties 4 & 5**	**Number w/ Partial Overlap**	**VI Metric**
1	No	No	No	0	0.7170
2	No	No	Yes	1	0.7729
3	No	Yes	No	1	0.7525
4	No	Yes	Yes	2	0.8085
5	Yes	No	No	1	0.7631
6	Yes	No	Yes	2	0.8191
7	Yes	Yes	No	2	0.7986
8	Yes	Yes	Yes	3	0.8546

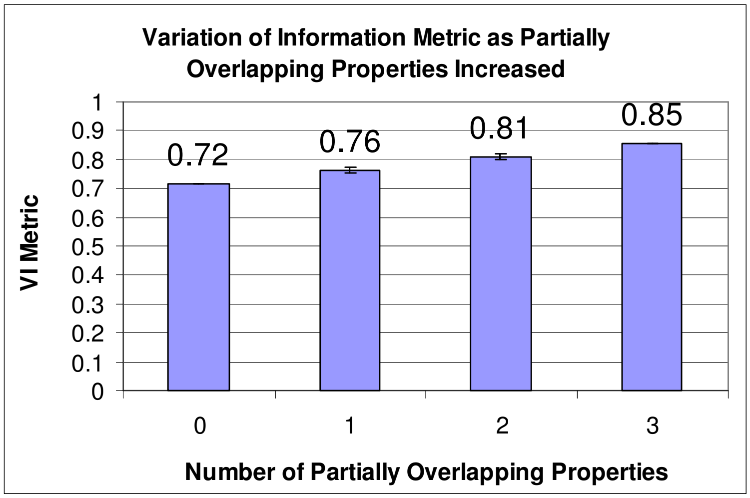


Figure 71 – This figure shows how the variation of information metric varies as the number of partially overlapping properties increases. As hypothesized, the metric increased, signifying greater non-overlap or loss of information.

6.4.3. Results and Discussion

Table 31 shows the raw data for the variation of information (VI) metric for each condition. As the number of partially overlapping properties varied from zero to three, the metric increased as hypothesized, from 0.72 to 0.85. The results are summarized as a graph in Figure 71. It shows how the variation of information metric varies as the number of partially overlapping properties increases. As hypothesized, the metric increased, signifying greater non-overlap or loss of information going from one clustering to the other.

Table 32 summarizes this experiment. The results of this experiment show that the variation of information metric behaves as hypothesized. We controlled for the amount of overlap between property representations of the two robots. This was possible because the properties were trained in a supervised manner, and hence the supervised training

Table 32 – Experimental summary and conclusions for the experiment regarding the calculation of the variation of information .

Experiment 9: General Experiment Summary **Calculating the Variation of Information Metric**	
Purpose	To determine whether the variation of information metric increases as the degree of overlap between properties increases.
Experiment Type	Real-robot (configuration 2)
Hypothesis	We hypothesize that the variation of information metric will increase when more properties overlap only partially.
Procedure	1. Train properties using labeled data for both robots(algorithm in Table 5). The training regime for the Amigobot was changed such that instances from two properties were joined. 2. Obtain instances from a shared context (Section 4.4) 3. Compute variation of information metric for different numbers of partially overlapping properties (algorithm in Table 29).
Independent Variable	The number of partially overlapping properties.
Dependent Variable	The value of the variation of information metric.
Conclusion	Hypothesis is confirmed. As the number of partially overlapping properties increased, so did the variation of information metric.

regime was able to be changed to include instances for two properties at a time as opposed to one. Such a training regime led to larger Gaussians (as seen in Figure 69) for one robot while the property Gaussians for the other robot remained the same. Now that we have verified that the metric can accurately reflect the amount of overlap between properties, we present in the next subsection results that show that the metric correlates inversely with the performance of knowledge transfer between the two robots.

6.5. Experimental Results: Correlating the V I Metric and Transfer Performance

In the previous experiment (Section 6.4), we have shown that the VI metric increases as the number of partially overlapping properties increases. This establishes that the information loss of going from one clustering to the other, as measured in an information-theoretic sense (Section 6.2), increases between the two sets of property representations. The purpose of this metric, however, is to be able to estimate the ultimate classification performance of actual knowledge transfer between the two robots. In other words, in order for the metric to be useful it must correlate with the actual classification performance. In theory, this should be the case, since as the amount of overlap between properties increases the membership values for those properties would increasingly differ between the two robots. However, it is important to validate that this does indeed occur on real robots using real-world data that is noisy.

6.5.1. Hypothesis

The hypothesis of this experiment is that the variation of information metric will inversely correlate with the efficacy of knowledge transfer, as measured by classification performance by the receiving robots. In other words, that the theoretical information loss measure will indeed signify loss of classification performance.

6.5.2. Procedure

The procedure of this experiment begins with the properties used in the previous experiment (Section 6.4). Recall that the experiment used a modified training regime for the Amigobot robot, where property pairs were merged into one representation. This was done for three pairs (six properties total). Eight conditions were used, where each condition specified whether the property representations for each property pair remained separate (i.e. unmodified as used in Sections 3.8.4, 4.5.3, and 5.7.3). Unlike the previous experiment in Section 6.4, after the properties were trained we then performed a second training process to learn concepts, as described in 3.5 using the algorithm in Table 6. In this case, we trained the first 26 of the original 34 concepts used in prior sections (Sections 3.8.4, 4.5.3, and 5.7.3). For the Pioneer robot, all of the properties remained the same and hence the concepts were the same as in prior experimental sections for this robot configuration. After the concepts were trained, they were transferred from the Amigobot robot to the Pioneer robot for all eight conditions (shown in Table 30).

For each condition, the performance over all concepts was measured in addition to the variation of information metric (shown in Table 31). The hypothesis of the experiment is that the two will inversely correlate; that is, as the variation of information metric increases (representing smaller overlap between properties and hence more information loss), classification performance will decrease. Just as before, we measure performance using receiver operator curves (ROC), recall rates, and precision rates. The ROC plots show the true positive ratio against the false positive ratio. The true positive ratio, or sensitivity, of a binary classifier is the number of true positives divided by the total number of positives in the test set. The false positive ratio is the number of false positives divided by the total number of negatives in the test set. The area under this curve represents overall accuracy over all of the objects. The recall rate measures the number of true positives divided by the number of positives in the test set, while precision measures the number of true positives divided by the number of test set instances that were classified to be positive. These numbers are proportions and can, if desired, be represented as percentages.

Table 33 – The VI metric and raw performance data (recall, precision, and area under the ROC curves). The VI metric and performance data inversely correlated, as verified by Pearson's correlation (Table 34).

Condition	Number w/ Partial Overlap	VI Metric	Recall	Precision	ROC Area
1	0	0.7170	0.8237	0.7641	0.7404
2	1	0.7729	0.7897	0.7293	0.7120
3	1	0.7525	0.7776	0.7237	0.7121
4	2	0.8085	0.7473	0.6908	0.6786
5	1	0.7631	0.7173	0.6641	0.6478
6	2	0.8191	0.7040	0.6492	0.6317
7	2	0.7986	0.7028	0.6613	0.6478
8	3	0.8546	0.6870	0.6412	0.6253

Given the VI metric and the three performance measures (area under the ROC curves, recall, and precision), we would like to measure the correlation between them. One method to perform this analysis is Pearson Product Moment Correlation, or Pearson's correlation coefficient for short (Snedecor & Cochran, 1980). This coefficient measures the degree of linear relationship between two random variables. It varies from -1 (indicating a strong inverse correlation) to 1 (indicating a strong direct correlation). Given two data sets (in our case the VI metric and performance metric for the eight conditions), the coefficient (r) can be measured as:

$$r = \frac{Cov(x, y)}{\sigma_x \sigma_y} \tag{15}$$

where $Cov(x, y)$ is the sample covariance of the two sets and σ_x and σ_y are the sample standard deviations of the data sets x and y, respectively.

Table 34 – The correlation between the variation of information metric and respective performance measure. High negative values, as shown, indicate strong negative correlation between the two random variables. P-values shown indicate that these correlations are statistically significant.

	Pearson's Correlation	P-value
Recall	-0.8374	0.0095
Precision	-0.8294	0.0109
Area	-0.828	0.0111

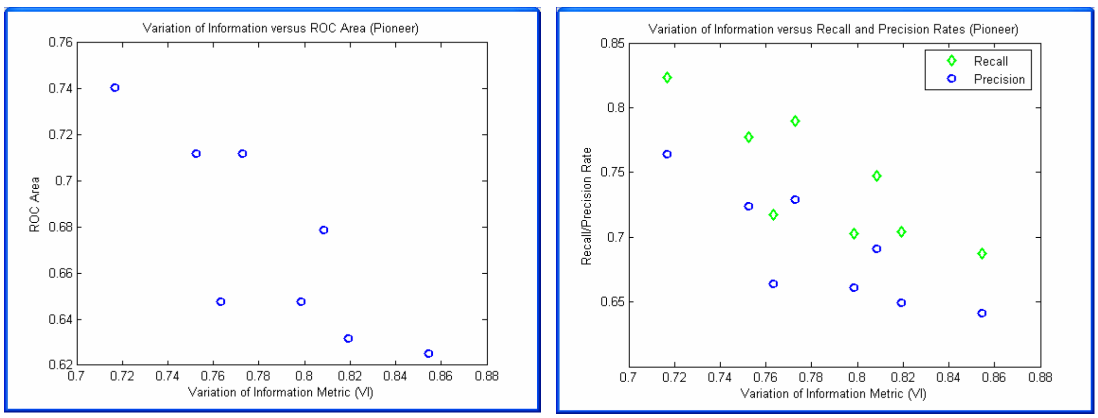


Figure 72 – This figure graphs the variation of information metric versus recall, precision, and area under the ROC curves when the receiving robot (Pioneer) classified concepts using test data. An inverse correlation is apparent, verified using Pearson's correlation in Table 34.

6.5.3. Results and Discussion

Table 33 shows the raw data for the recall, precision, and area under the ROC curve for the eight conditions. It also reproduces the VI metric from Table 31. Figure 72 plots the performance data against the corresponding VI metric. The two data sets inversely correlated, as hypothesized. In order to quantitatively verify this, we show the Pearson's correlation values for the three metrics in Table 34. The results show an inverse correlation (-0.84, -0.83, and -0.83 for recall, precision, and ROC area, respectively) between the VI metric and knowledge transfer performance. P-values are shown to prove statistical significance (p-value equals 0.0095, 0.0109, and 0.0111 for recall, precision, and ROC area, respectively), where the values indicate the probability of getting the correlation observed by random chance.

Table 35 – Experimental summary and conclusions for the experiment regarding the correlation between variation of information and knowledge transfer effectiveness.

Experiment 10: General Experiment Summary **Correlating Variation of Information and Transfer Learning Performance**	
Purpose	To determine whether the variation of information metric correlates with the effectiveness of knowledge transfer.
Experiment Type	Real-robot (configuration 2)
Hypothesis	We hypothesize that the variation of information metric will inversely correlate with the resulting performance of knowledge transfer. In other words, as the variation of information increases, knowledge transfer effectiveness will decrease.
Procedure	1. Train properties using labeled data for both robots(algorithm in Table 5). 2. Train concepts using labeled data for both robots (algorithm in Table 6). 3. Transfer concepts from one robot to the other (algorithm in Table 22). 4. Classify concepts for test data (Table 4). 5. Measure classification performance on receiving robot on test set.
Independent Variable	The number of partially overlapping properties.
Dependent Variable	The value of the variation of information metric and the classification performance after knowledge transfer.
Metric	Pearson's Correlation, which measures the statistical relationship between two random variables.
Conclusion	Hypothesis is confirmed. The variation of information inversely correlated with knowledge transfer effectiveness in a statistically significant way.

Table 35 summarizes this experiment. The results of this experiment demonstrate an important characteristic: The information loss metric, as measured using information theory, inversely correlates with actual classification performance on real robots using real noisy data. In other words, the robots can rely on the information loss metric to reflect the performance of knowledge transfer. This indicates that the robots may be able to use the metric to estimate the efficacy of knowledge transfer *a priori* (i.e., before actual transfer occurs). We will show in the next chapter that this is indeed the case, and can be used to pick robots for knowledge transfer in a manner that maximizes knowledge transfer performance.

6.6. Summary

This chapter dealt with heterogeneity types in class H2b and H3b, where robots may have partially overlapping properties. While previously (e.g. in Chapter 4) we treated properties as matching or not matching, here we treat their overlap in a continuous manner. In Section 6.1, we discussed how such partial overlaps may be a source of error during knowledge transfer between two robots. We then introduced the notion in Section 6.2 that information theory can aid the robots in determining how much information is lost when such partial overlaps occur. Specifically, the variation of information (VI) metric, used in the clustering community to measure information loss when going from one clustering to another (Meila, 2002), can be used to measure the amount of information loss (non-overlap) between candidate property pairs. We detailed the algorithm for measuring this metric using real data in Section 6.3.

We tested two hypotheses in this chapter regarding the variation of information metric:

1) We hypothesized, and confirmed the hypothesis, that the metric will indeed increase as the number of partially overlapping properties increases (Section 6.4). Since the training data for the properties consists of supervised labels, we were able to modify the training for one of the robots to merge pairs of properties together, resulting in

larger properties than the other robot. As hypothesized, as the number of properties pairs that were merged increased, the variation of information metric increased as well. This demonstrated that the metric could indicate the amount of overlap between properties.

2) We hypothesized, and confirmed the hypothesis, that the variation of information metric will inversely correlate with knowledge transfer efficacy (Section 6.5). That is, as the properties overlapped to a lesser extent, knowledge transfer using these properties will become less effective. We confirmed the hypothesis by transferring concepts that used the modified (partially overlapping) properties and showed that the variation of information metric inversely correlated with classification performance measured using recall, precision, and area under the ROC curves. Correlation was measured using Pearson's correlation coefficient, and the correlations were shown to be statistically significant.

In summary, we have shown that the VI metric can measure the amount of overlap between properties and that this characteristic inversely correlates with knowledge transfer performance. This suggests that robots may be able to take into account property overlaps using this metric to estimate the performance of knowledge transfer a *priori*. Recall that we have previously discussed methods for accounting for the other source of knowledge transfer error (missing properties on one robot and their relative importance) in Section 5.8. In the next chapter, we combine both of these methods to pick a robot from a set of robots such that the knowledge transfer performance is maximized. We also demonstrate other methods of communication, such as picking a distinguishing property that can distinguish a concept from a set of concepts. These methods combine technique and results that we have discussed thus far in this dissertation.

CHAPTER 7

UTILIZING LEARNED SIMILARITY MODELS FOR COMMUNICATION AND COORDINATION

7.1. Utilizing Similarity Models for Various Types of Communication

The preceding two chapters (5 & 6) dealt with transfer of an entire concept, facilitated by the property mapping models discussed in Chapter 4 and the variation of information metric models discussed in Chapter 6. There are several other useful ways in which both of these similarity models can facilitate communication and task performance. We now turn our attention to these practical applications, including various scenarios in which these models can be used to pick the distinguishing properties of a concept that will be understood by the receiving robot and to pick the concept among a set of concepts that are more likely to be classified well based on the information loss measure.

7.2. Choosing Distinguishing Properties

One possible type of communication that will be required in a search and rescue scenario (among others) is a description of a particular instance of a concept that can distinguish it from a surrounding context. For example, a robot with CO_2 sensors can describe the victim that is still living based on perceptual features that another robot (that may not have that sensor) can distinguish. This can also be useful in cue-based navigation where directions such as ‘turn right at the blue box’ can be given.

Let $P_c^{A,B} = P_c^A \cap P_c^B$ be the set of shared properties involved in robot A’s representation for concept c (this can be the matrix representation for the concept, or the

Table 36 - Algorithm for determining a distinguishing property of a concept from a set of concepts.

Algorithm: Choosing Distinguishing Properties

Input: A set of concepts C' and a target concept c
Properties P^A , Properties P^B , PropertyMapping *PM*,
Output: Value and property number of a distinguishing property

```
// Note: PropertyMapping PM consists of an array where PM[ p ]
// contains the index to the property p' on robot A that maps to
// property p on robot B.  These mappings are obtained using the
// methods in Chapter 4.

// Obtain diagonals of surrounding concept matrices (if instances are used
// as opposed to abstract concept, this is not necessary).
PropertiesOfInstances = Diagonals of all concepts matrices in C'

PropertiesOfTarget = Diagonal of c

// Find maximal value for each property (Step 1)
MaxPerProperty = max(PropertyOfInstances)

// Find complement of each value (one – value) (Step 2)
ComplementPerProperty = 1 – MaxPerProperty

// Intersect (min operation) with properties of c (Step 3)
IntersectionPerProperty = min(PropertiesOfTarget, ComplementPerProperty)

// Find maximal value, as long as it is in the PropertyMapping model (Steps 4 & 5)
DistinguishingProperty = argmax(IntersectionPerProperty) for all properties in PM

Return DistinguishingProperty
```

conversion of a particular instance into the matrix representation, as described in Section 3.4). Now suppose that there is a set of concepts *C'* consisting of concepts from which *c* must be distinguished (either because they are nearby, or in the case of cue-based navigation in the same environment). The goal is to provide a distinguishing property that can distinguish concept *c* from its surroundings. Of course, the robots must first build a model of which properties are shared or unshared (Chapter 4) in order to avoid using properties that the other robot does not have.

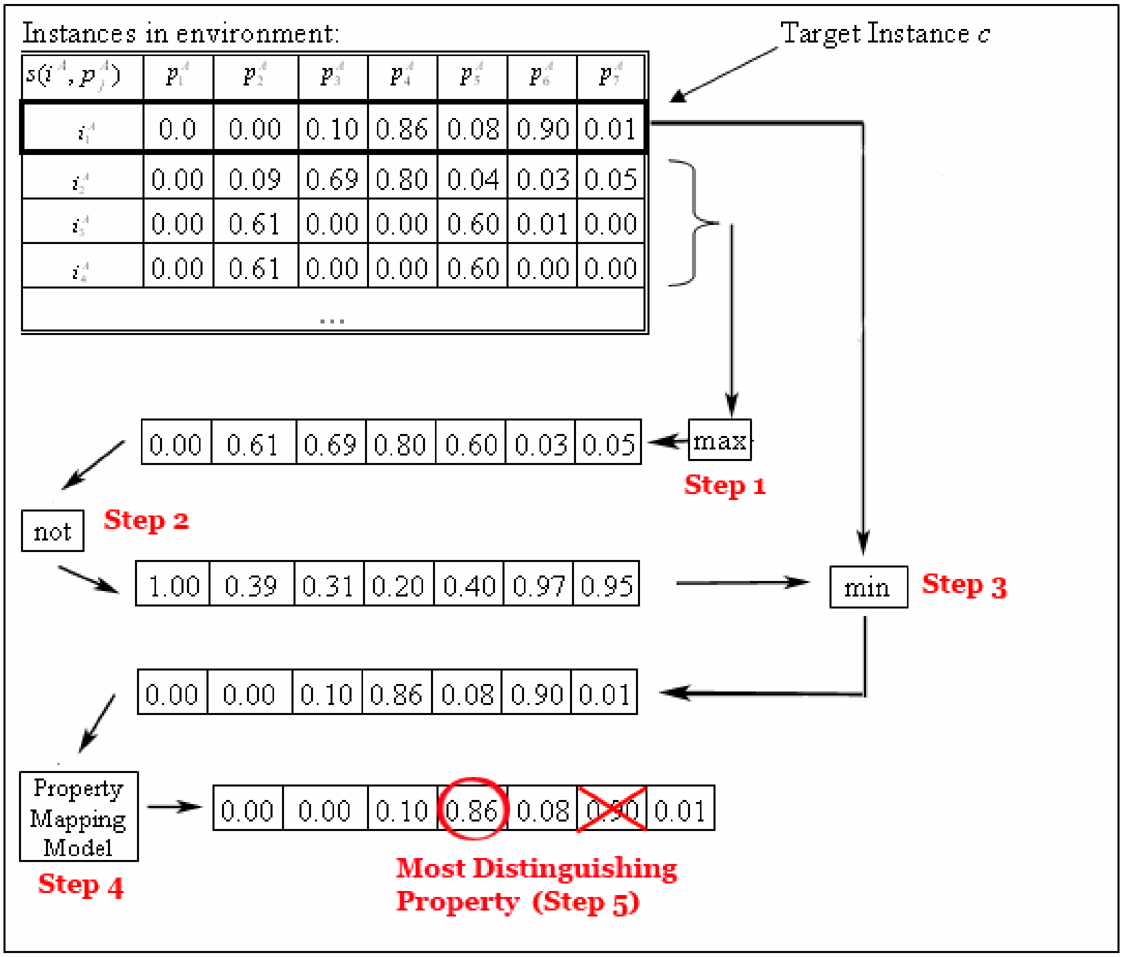


Figure 73 – Example demonstrating choice of a property distinguishing the target instance from other surrounding instances.

Table 36 shows the algorithm for doing this, and Figure 73 shows an example. We begin by finding the union of all shared properties present in the concepts C' (Step 1). If properties are continuous, as they are in this work, the set operation can be performed using the *max* operator. If concept matrices are used (representing abstract concepts), the property memberships are located in the diagonal of the concept matrix. For distinguishing actual instances of objects, the property memberships for each instance can be used directly. The max operator is then performed on these values across all concepts in C'. We call the resulting property memberships $P_{C'}$. These values represent properties that have high membership in at least one of the surrounding concepts. This being the case, even if the target concept *c* has a strong membership in these properties, they cannot be used to distinguish the target concept (e.g., if some of the surrounding objects have high memberships in the "blue" property, then this color property cannot be used to distinguish the target concept).

The second step is to take the complement of $P_{C'}$; in fuzzy logic one can subtract the original memberships from one (Step 2). High values in the complement of $P_{C'}$ represent properties that are *not* high in the surrounding concepts (e.g. none of the objects are "green"). High values for the target concept for these properties can be then used to distinguish it.

The properties of the target concept *c* can then be intersected with the complement of $P_{C'}$ (Step 3). This results in properties that have high memberships for the target concept but *not* the surrounding concepts. Figure 73 shows an example for finding such a property. Properties with high memberships represent distinguishing properties. Properties that are not shared are then removed from potential properties to choose from (Step 4). Since these are taken from only the shared properties, robot B will be able to distinguish concept *c* using any of the remaining properties. In order to estimate the most distinguishing property, the property with the maximal value can be used (Step 5). Once the most distinguishing property is determined, it is then transferred to the receiving robot. Using the similarity or property mapping model (built using the algorithms and methods in Chapter 4), the receiving robot can then find the concept whose distinguishing property is closest to the value received from the other robot. Figure 74 summarizes the procedure. We will now describe experiments validating this approach.

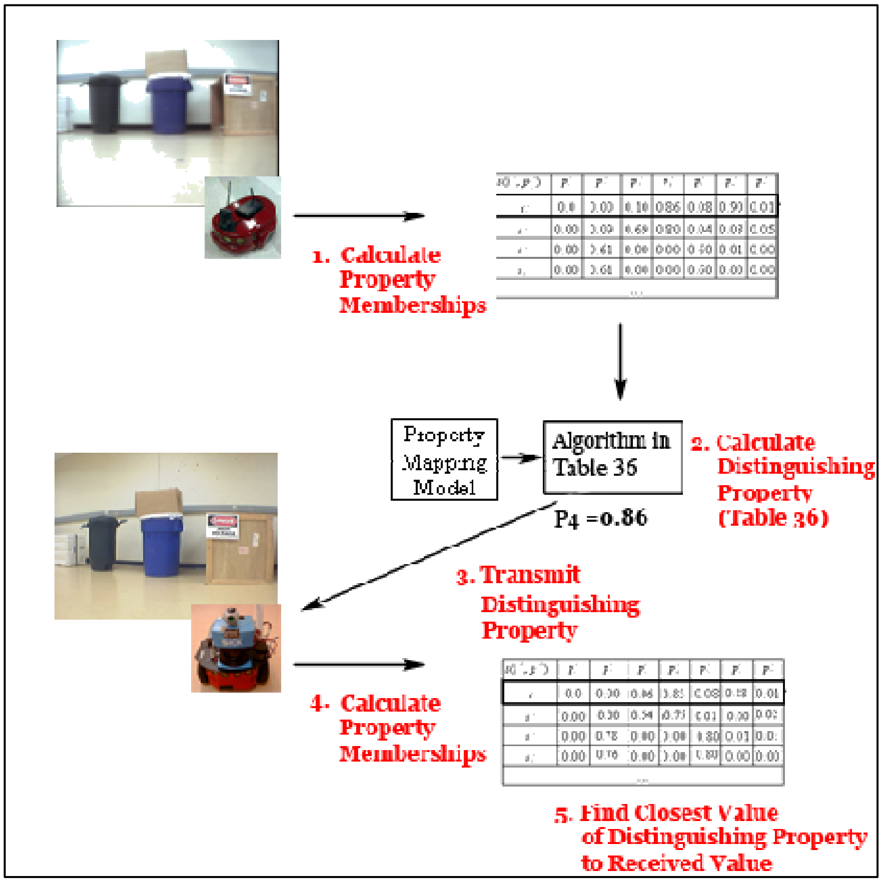


Figure 74 – Example demonstrating the steps involved in the procedure of the experiment. First, the Amigobot calculates the property memberships, using properties it learned. Using the algorithm in Table 40, the Amigobot then picks a distinguishing property and sends it (along with the membership value for that property) to the Pioneer. The Pioneer robot then calculates property memberships using its own sensors and properties, and finds the closest one to the distinguishing property received from the Amigobot. The resulting concept is the estimated target, and can be compared to ground truth.

7.2.1. Experimental Hypothesis

We hypothesize that the robot will be able to successfully find distinguishing properties (and in particular the most distinguishing property), taking into account shared properties and utilizing fuzzy logic, that the receiving robot can then use to distinguish a concept accurately from surrounding concepts. Specifically, we hypothesize that when the receiving robot uses the most distinguishing property, performance in terms of choosing the target concept will be better than when choosing a random concept or when using a particular property across all trials.

7.2.2. Experimental Procedure

In order to test the hypothesis, we first trained both robots as in Section 3.8.4. One robot (the *expert robot*) first learned concept matrices for a set of concepts, and built similarity or property mapping models with the other robot based on the procedures described in the previous chapter (Section 4.2.1 and specifically algorithm in Table 16). In this case, the expert robot was the Amigobot while the receiving robot was the Pioneer. From the resulting thirty-four concepts, three random concepts were then chosen. One of them was randomly designated as the target concept, while the other concepts were designated as the surrounding concepts.

The expert robot then used the algorithm in Table 36 to pick a distinguishing property. The receiving robot then determined which of its properties corresponded to the distinguishing property (based on the property mapping model), and then determined which concept had a similar membership as the received value for this distinguishing property. This concept is what the receiving robot has estimated to be the target concept. This chosen concept was then compared to the ground truth target concept, and accuracy was determined over 100 such trials. Accuracy was measured by the number of times the target concept was chosen correctly, divided by the number of trials. The entire process was then repeated ten times, with means and standard deviations being measured. As a means of comparison, the accuracy of picking out the target concept using one particular property across all 100 trials was measured as well.

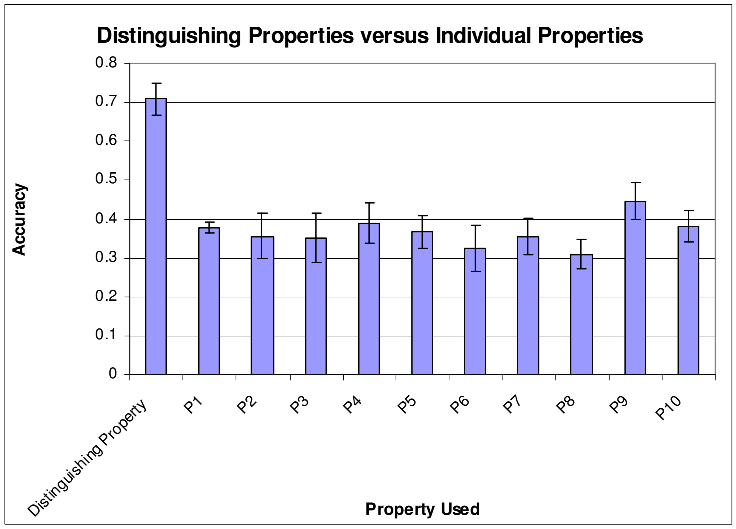


Figure 75 – Graph showing the accuracy of the receiving robot in locating the target concept from two other concepts. The results are averaged over all ten trials, each consisting of a hundred situations. The accuracy of the algorithm in Table 36 is shown on the left ("Distinguishing"), while the rest of graph shows the accuracy when one particular property is used across all trials. The algorithm was able to correctly pick out the target concept significantly better than choosing any particular property (rate of 0.71 versus maximum of 0.45) as well as better than chance, which would achieve an accuracy of 0.33.

7.2.3. Experimental Results and Discussion

Figure 75 shows the results of the accuracy results averaged over ten trials (each consisting of one hundred trials). Using the algorithm ("Distinguishing Property"), the receiving robot was able to distinguish the target concept 71% of the time, significantly better than when using any fixed property across all instances (whichachieved a maximum accuracy of 45%) as well as random chance (which would achieve 30% accuracy). These results show that the concept representation in conceptual spaces allows a robot to pick distinguishing properties. Unshared properties can be ruled out using the similarity models developed in Chapter 4. In other words, the models of which properties are shared are useful not just for knowledge transfer, but also for various communicative acts such as this one. Table 37 summarizes this experiment.

Table 37 – Experimental summary and conclusions for the experiment regarding choosing a distinguishing property.

Experiment 11: General Experiment Summary **Choosing Distinguishing Properties**	
Purpose	To determine whether a robot can use the property mapping model and conceptual spaces representation to choose a distinguishing property that distinguishes one concept from a surrounding set of concepts.
Experiment Type	Real-robot (configuration 2)
Hypothesis	We hypothesize that the robot will be able to successfully find distinguishing properties, taking into account shared properties and utilizing fuzzy logic, that the receiving robot can then use to distinguish a concept from its surroundings.
Procedure	The following procedure was run ten times, each of which consisted of 100 situations: 1. Train properties using labeled data for both robots(algorithm in Table 5) 2. Train concepts using labeled data for both robots (algorithm in Table 6) 3. Set up a situation with three concepts, one designated as the target. 4. Expert robot determines the most distinguishing Property (Table 36) 5. Send the distinguishing property to the receiving robot 6. The receiving robot determines which of its properties corresponds to the distinguishing property (based on the property mapping model), and then determines which concept has a similar membership as the received value for this distinguishing property.
Metric	Accuracy as measured by the number of times the receiving robot correctly picks out the target concept divided by the total number of trials.
Conclusion	Hypothesis is confirmed. The receiving robot was able to correctly distinguish the target concept at a significantly better percentage than chance or when using a fixed property across all trials.

7.3. Favoring a Concept Among a Set of Concepts

Another type of communication that would be useful is for one robot to pick a concept from among a set of concepts. This would be useful, for example, for describing a particular room based on objects that are in it. The objective is for robot A to estimate which concept will maximize the chance that robot B will detect it. This can be done using the models of robot differences learned, including the property mapping model (algorithm in Table 16) as well as the variation of information metric (algorithm in Table 29). Given these models, robot A can find the concept that has the minimal amount of variation of information in shared properties that are involved in the concept.

Table 38 describes the algorithm. The algorithm is given as input a set of concepts to choose from and the variation of information metric between all property pairs shared by the robots (built using methods in Chapter 6, specifically the algorithm in Table 29). The goals is to choose one concept from the set, the classification rate of which should be higher than the other concepts. In practice, the robot that will be transferring concepts (the *expert* robot) to the other robot can only estimate the resulting performance. It does this by calculating, for each concept in the set, the summation of variation of information for all property pairs used in the concept (i.e. the degree of non-overlap between the properties). This is combined with the actual property values along the diagonal of the concept matrix for the respective properties of the concept. The idea is that properties that have low memberships for the concept are less important, and hence the variation of information metric for that property is not as important as for highly-valued properties. In this dissertation, we use a simple combination where the two values are multiplied. Other more sophisticated combination methods, for example utilizing machine learning, are possible.

In addition, we further combine this estimate with the one obtained as was detailed in Section 5.8. Specifically, the expert robot performs classification using its learned

Table 38 - Algorithm for picking a concept from a set of concepts that maximizes the receiving robot's classification accuracy.

```
Algorithm: Favoring a Concept Among a Set of Concepts

Input: A set of concepts C , Performance Estimates E
        Properties P^A , Properties P^B , PropertyMapping PM, Variation of Information VI
Output: Value and concept number of best estimated concept

// Obtain diagonals of surrounding concept matrices (if instances are used
// as opposed to abstract concept, this is not necessary).
PropertiesOfInstances = Diagonals of all concepts matrices in C'

MaxConceptIndex = 0
MaxConceptEstimate = -Infinity
For each concept c in C'
   // Obtain property memberships in c for all properties
   PropertiesOfTarget = Diagonal of c

   VI_sum = 0
   IncrementCount = 0

   // Find average VI for all shared properties used in the concept, weighted by
   // actual membership in concept
   For each property p1 in P^A
     // Property is not shared between the two robots
     If PM[p1] < 0
        // Do nothing
     Else
        VI_sum = VI_sum + VI( PM[p1], p1) * PropertiesOfTarget(p1)
        IncrementCount = IncrementCount + 1
     End
   End

   // Divide by total number of VI values used (to obtain average)
   VI_sum = VI_sum / IncrementCount

   // Combine VI with performance estimate (from Section 5.8)
   CombinedValue = E[c] / VI_sum
   If (CombinedValue > MaxConceptEstimate)
     MaxConceptIndex = c
     MaxConceptEstimate = CombinedValue
   End
  End
End

Return {MaxConceptIndex, MaxConceptEstimate}
```

models with the unshared properties removed. This estimate is then divided by the average variation of information metric over shared properties involved in the concept. In other words, the estimate is downgraded if the corresponding properties only partially overlap. Again, future work can involve the development of a more sophisticated combination method. We will now describe experiments validating this capability.

7.3.1. Experimental Hypothesis

The hypothesis is that a robot can effectively use the learned similarity models (confusion matrices learned as per the algorithm in Table 16 and the variation of information metric learned as per the algorithm in Table 29) to describe aspects of the world in a manner that maximizes understandability by the receiving robot. Specifically, we hypothesize that using this information, the transferring (expert) robot can pick a concept that will be more accurately classified by the receiving robot than picking a concept at random.

7.3.2. Experimental Procedure

In order to demonstrate the efficacy of the methods, we will build on the experiments in the previous subsection. Specifically, the experiment will start with two heterogeneous robots, one of which learns a set of concepts. The two robots also learn models of their differences in the form of confusion matrices and variation of information metric, as described previously (algorithms in Table 16 and Table 29, respectively). One robot, the *expert*, will pick a concept from a set of concepts based on the methods described above (algorithm in Table 38). The other robot is the *novice* robot. This robot will have some overlapping set of properties and concepts, with varying degrees of differences. In this case, the expert robot was the Amigobot while the novice robot was the Pioneer. From the resulting thirty-four concepts, ten random concepts were then chosen as the set of concepts to choose from. The expert robot then chose one concept from this set based on the algorithm described in Table 38. The classification performance of the chosen concept by the novice robot was then recorded, and the average performance over one hundred of these trials was measured. Performance was again measured using recall rates, precision rates, and the areas under the ROC curves. The conditions for this experiment consisted of a control in which the expert robot only picked concepts randomly, and the experimental condition where the expert robot took into consideration the capabilities of the novice robot. The performance of choosing the best possible concept overall was also recorded for comparison.

We expect that when the expert takes into account the capabilities of the novice robot, the performance of the novice robot will be better than when choosing randomly. Performance in this case will be measured as the ability of the novice robot to distinguish concepts using the properties given to it by the expert robot, and to detect particular concepts given to it by the expert robot.

7.3.3. Results and Discussion

Table 39 shows the results. The "Random" column shows the performance metrics (averaged over all 100 trials) by the receiving robot for a randomly-chosen concept. The "Estimate" column shows the performance metrics for the concept chosen by the expert robot using the algorithm in Table 38. The results for the latter (precision rate of 0.75, recall rate of 0.77, and ROC area of 0.74) is significantly better than for a randomly-chosen concept (precision rate of 0.68, recall rate of 0.71, and ROC area of 0.67). The differences are significant, as shown in the "P-Value" column, which were less than 0.05 for all metrics. Finally, the last column shows the best possible performance that can be achieved (0.90 precision rate, 0.95 recall rate, and 0.89 ROC area). Although the use of the algorithm resulted in performance that was significantly better than a randomly-chosen concept, it did not achieve rates equivalent to the optimal choice. As stated earlier, the estimated performance values calculated by the expert robot are only estimates, and could be off due to noise or other factors. Also, as mentioned, more sophisticated methods of combining the variation of information metric, the importance of a property to a concept, and the performance estimates could yield significantly better results.

Table 39 - Results demonstrating the advantage of choosing a concept from a set based on models of robot differences. The "Random" columns shows the three performance metrics for when the expert robot randomly chose a concept. The "Estimate" column shows the results when the robot chose a concept using the algorithm. Significantly higher average performance is achieved when using the algorithm. Statistical significance is shown in the "P-Value" column which shows the significance of differences between using the algorithm and randomly choosing a concept. The final column to the right shows the best possible performance that can be achieved.

	Random		Estimate			Best
	Mean	**Stdev**	**Mean**	**Stdev**	**P-Value**	**Mean**
Precision	0.681	0.186	0.747	0.153	0.007	0.905
Recall	0.716	0.212	0.771	0.164	0.042	0.946
ROC Area	0.668	0.181	0.742	0.156	0.002	0.886

Table 40 summarizes this experiment. In this experiment, we showed that information obtained by combining the performance estimates (obtained as described in Section 5.8) as well as the variation of information metric can result in significant gains when a robot needs to pick a concept from a set of concepts such that the receiving robot will classify it well. This shows the first evidence of the usefulness of the information-theoretic metric described in Chapter 6. In that chapter we showed that the metric correlated with knowledge transfer, and here we show that such correlations can be leveraged during decision making and communication. In the next section, we show further evidence of this, in the form of a robot being able to pick the most similar robot to itself using VI.

Table 40 – Experimental summary and conclusions for the experiment regarding choosing the best-recognized concept from a set of concepts.

Experiment 12: General Experiment Summary and Conclusion **Favoring a Concept Among a Set of Concepts**	
Purpose	To determine whether a robot can use the property mapping model and variation of information to choose a concept from a set of concepts such that the classification performance of the receiving robot is increased compared to randomly choosing a concept.
Experiment Type	Real-robot (configuration 2)
Hypothesis	We hypothesize that the robot will be able to successfully choose a concept based on models of robot differences, and that the performance on the receiving robot will be better for this concept than for a randomly chosen concept.
Procedure	1. Train properties using labeled data for both robots(algorithm in Table 5) 2. Train concepts using labeled data for expert robot (algorithm in Table 6) 3. Choose ten concepts from the thirty-four at random. 4. Determine the best estimated concept (algorithm in Table 38). 5. Transfer all ten concepts to receiving robot. 6. The receiving robot classifies all ten concepts using the received representation. The performance on the chosen concept is compared to a randomly-chosen concept.
Metric	Accuracy of classification, as measured by recall rate, precision rate, and area under the ROC curve.
Conclusion	Hypothesis is confirmed. The receiving robot was able to classify the chosen concept significantly better than a concept chosen at random.

7.4. Speaking to Like-Minded Robots: Measuring Levels of Conceptual Discordance

A final way that we utilize the similarity models and consider heterogeneity is in the decision of *which peer* to exchange knowledge with. We use both the estimation of performance based on shared properties (Section 5.8) as well as the variation of information (VI) metric (Section 6.2). Specifically, we use the former method to estimate the performance that would result after knowledge transfer. This estimate is then multiplied by the variation of information metric subtracted from one. This downgrades the performance estimate by the amount of non-overlap between the shared properties. This is done for all potential robots, and the robot with the maximal estimated performance is chosen to transfer knowledge to the receiving robot. Table 41 shows the algorithm.

7.4.1. Experimental Hypothesis

The goal of this experiment is to show that a measure combining the two methods of knowledge transfer estimation can be successfully used to choose robot partners in order to increase the effectiveness of the communication. It also further demonstrates the usefulness of the learned models of robot similarities and differences. The hypothesis is that the variation of information metric and *a priori* estimation based on unshared properties can be used to successfully choose a robot partner. In other words, that robot partners chosen using these methods will result in higher after-transfer performance compared to choosing a robot at random or without incorporating the VI metric.

Table 41 - Algorithm for picking a robot from a set of robots in order to maximize post-transfer classification accuracy.

```
Algorithm: Picking the Most Similar Robot

Input: A set of concepts C, Performance Estimates E, a set of robots R, VI weight w
       Properties P^A, Properties P^B, PropertyMapping PM, Variation of Information VI
Output: Most similar robot r

// Obtain diagonals of surrounding concept matrices (if instances are used
// as opposed to abstract concept, this is not necessary).
PropertiesOfInstances = Diagonals of all concepts matrices in C'

MaxRobotIndex = 0
MaxRobotEstimate = -Infinity
For each robot r in R
   // Obtain property memberships in c for all properties
   PropertiesOfTarget = Diagonal of c

   VI_sum = 0
   IncrementCount = 0

   // Find average VI for all shared properties
   For each property p1 in P^A
     // Property is not shared between the two robots
     If PM[p1] < 0
       // Do nothing
     Else
       VI_sum = VI_sum + VI( PM[p1], p1)
       IncrementCount = IncrementCount + 1
     End
   End

   // Divide by total number of VI values used (to obtain average)
   VI_sum = VI_sum / IncrementCount

   // Combine VI with performance estimate (from Section 5.8)
   CombinedValue = E[r] * ( (1-VI_sum) * w)
   If (CombinedValue > MaxRobotEstimate)
     MaxRobotIndex = r
     MaxRobotEstimate = CombinedValue
   End
  End
End

Return {MaxRobotIndex, MaxRobotEstimate}
```

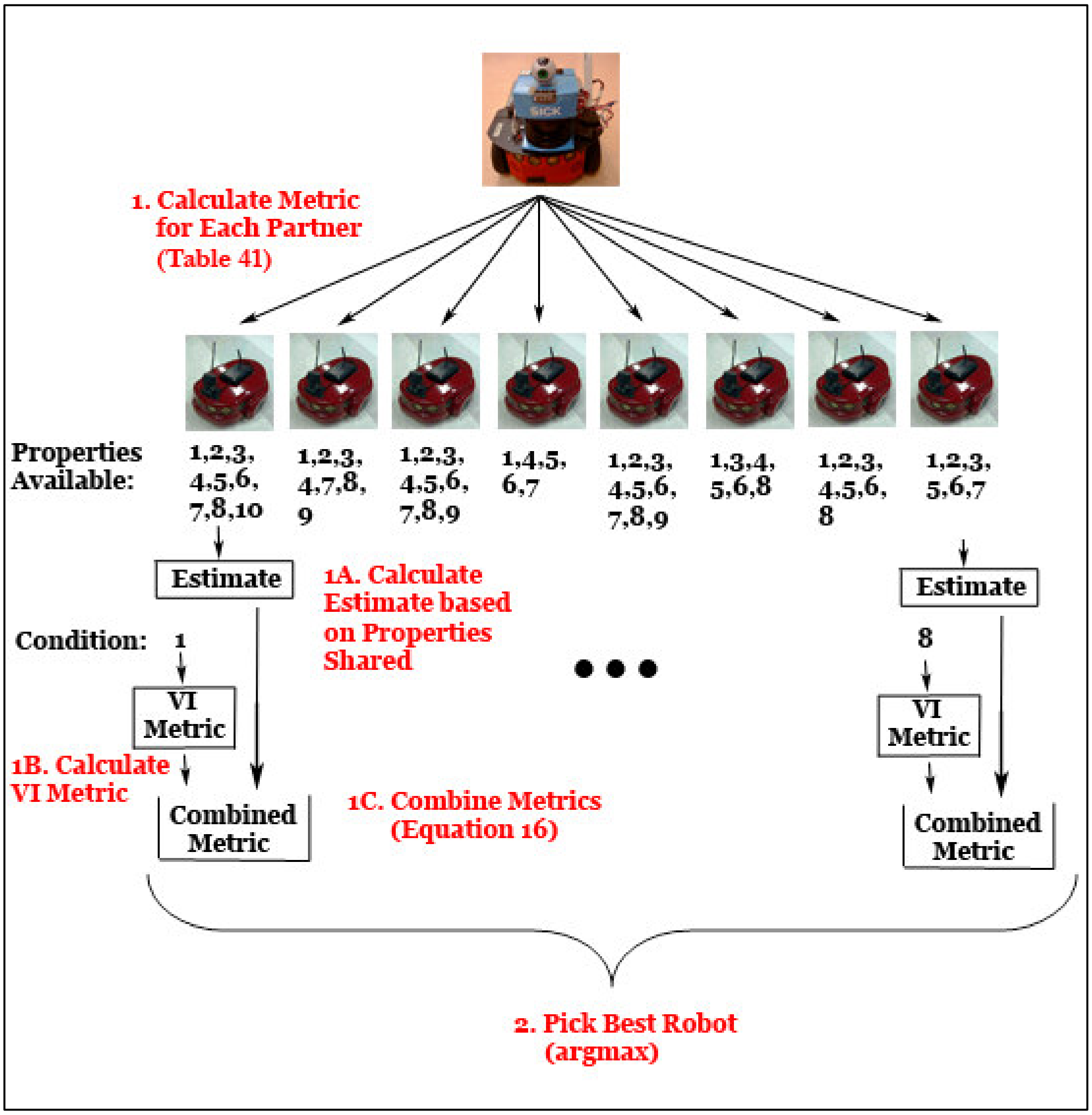


Figure 76 – The steps involved in the procedure of the experiment. First, the Pioneer robot calculates the metric for each partner (Step 1). This consists of calculating the *a priori* estimate (Step 1A) as was done in Section 5.8. The VI metric is also calculated, using the algorithm in Table 29 (Step 1B). The VI metric varies depending on the condition, which changes the Amigobot's properties, resulting in different amounts of overlap between the properties of the Amigobot and Pioneer. These two metrics are combined according to Equation 16. Finally, the best partner robot is chosen. Knowledge transfer efficacy can then be compared when using this best robot or a partner robot.

Table 42 – Experimental summary and conclusions for the experiment regarding picking the best robot from a set of robots.

Experiment 13: General Experiment Summary and Conclusion **Choosing the Best Robot for Knowledge Transfer**	
Purpose	To determine whether a robot can use the property mapping model and variation of information to choose a robot from a set of robots such that the classification performance of the receiving robot after knowledge transfer is better than picking a robot at random.
Experiment Type	Real-robot (configuration 2)
Hypothesis	We hypothesize that the robot will be able to successfully choose a most-similar robot based on models of robot differences, and that the performance on the receiving robot will be better for this robot than for a randomly chosen robot.
Procedure	1. Train properties using labeled data for both robots(algorithm in Table 5) 2. Train concepts using labeled data for expert robot (algorithm in Table 6) Build property mapping model (algorithm in Table 15) and variation of information metric (algorithm in Table 29) 3. Determine the best robot for transfer (algorithm in Table 41). 5. Transfer all concepts to all potential partner robots. 6. The receiving robot classifies all concepts using the received representation. The performance on the chosen robot is compared to a randomly-chosen robot.
Metric	Accuracy of classification, as measured by recall rate, precision rate, and area under the ROC curve.
Conclusion	Hypothesis is confirmed. The chosen receiving robot was able to classify the concepts significantly better than a randomly-chosen robot.

7.4.2. Experimental Procedure

Table 42 summarizes the experiment and Figure 76 depicts the procedure. In order to demonstrate the efficacy of the measure of heterogeneity, we will build on the experiments in the previous chapters. The procedure of this experiment begins with the properties used in the previous experiment (Section 6.4). Recall that the experiment used a modified training regime for the Amigobot robot, where property pairs were merged into one representation. This was done for three pairs (six properties total). Eight conditions were used, where each condition specified whether the property representations for each property pair remained separate (i.e. unmodified as used in Sections 3.8.4, 4.5.3, and 5.7.3). Unlike the previous experiment in Section 6.4, after the properties were trained we then performed a second training process to learn concepts, as described in Section 3.5 using the algorithm in Table 6. In this case, we trained the first 26 of the original 34 concepts used in prior sections (Sections 3.8.4, 4.5.3, and 5.7.3). For the Pioneer robot, all of the properties remained the same and hence the concepts were the same as in prior experimental sections for this robot configuration. After the concepts were trained, they were transferred from the Pioneer robot to the Amigobot robot. For each potential partner robot, the condition varied from one to eight (the eight conditions shown in Table 30). This determined how many partially overlapping properties the Amigobot had. Recall that as the number of partially overlapping properties increased, so did the variation of information metric (Section 6.4). Furthermore, as the number of partially overlapping properties increased the knowledge transfer performance decreased (Section 6.5).

In addition, a random subset of properties was chosen as shared between the two robots. Specifically, eight random subsets were used to represent eight random configurations of the Amigobot robot that the Pioneer robot could transfer knowledge to. Each trial used anywhere from five to ten randomly chosen properties to be shared. This diverges from the experiments in sections 6.4 and 6.5, and adds another source of potential error during knowledge transfer.

For each trial, the estimated performance over all concepts was measured in addition to the variation of information metric (shown in Table 31). The performance was estimated *a priori* by measuring the performance of the transferring robot (the Pioneer) using only the shared properties (as was done in Section 5.8). The performance estimate for each of the eight potential partner robots was then multiplied by one minus the variation of information metric (since higher VI or information loss would result in lower performance). Formally:

$$FinalEstimate = Estimate * (1 - VI) \qquad \textbf{(16)}$$

Note that other techniques for combining the two estimates are possible. For example, machine learning could be used to decide how best to combine this information to better predict the performance of the receiving robot. The maximal estimate value was then used to choose the best partner robot. The after-transfer performance was then measured (by testing the transferred representations on the Amigobot, as was done in Section 5.7.3). The entire process (with eight random partner robots) was then repeated twenty times, and the average actual achieved performance by the chosen robot was measured. This was compared to the results when choosing a random robot, as well as when using *only* the *a priori* estimation *without* using the variation of information metric. The hypothesis is that higher classification performance will be achieved when using both the *a priori* estimate as well as the VI metric compared to using estimation without the VI metric. We further hypothesize that both of the methods will result in better performance than picking a random partner robot out of the eight potential choices. Note that the methods in this section are applicable to both directions of transfer (determining who to transfer knowledge to as well as determining who to receive knowledge from). In this experiment the Pioneer robot determines who to transfer to.

Just as before, we measure performance using receiver operator curves (ROC), recall rates, and precision rates. The ROC plots show the true positive ratio against the false positive ratio. The true positive ratio, or sensitivity, of a binary classifier is the number of true positives divided by the total number of positives in the test set. The false positive ratio is the number of false positives divided by the total number of negatives in the test set. The area under this curve represents overall accuracy over all of the objects. The recall rate measures the number of true positives divided by the number of positives in the test set, while precision measures the number of true positives divided by the number of test set instances that were classified to be positive. These numbers are proportions and can, if desired, be represented as percentages.

7.4.3. Experimental Results and Discussion

To analyze the results, we begin by showing results for one of the twenty trials, consisting of the Amigobot choosing from a set of eight robots. Subsequently we will show results averaged over all trials. Table 43 shows the raw data for the eight randomly-chosen partner robots, including the "1-VI" condition, the *a priori* estimated performance ("Estimate"), the combination of *a priori* estimation and VI ("Estimate with VI"), and resulting knowledge transfer performance in the form of recall ("Transfer").

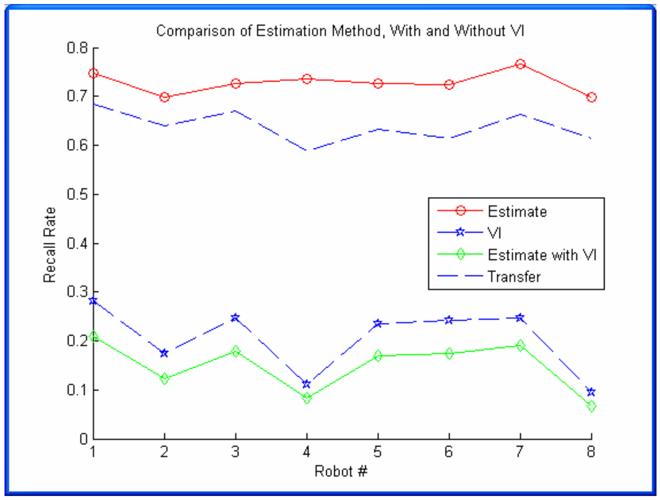


Figure 77 – Graph showing the recall rate after transfer to the eight partner robots. The "Estimate" values do not change much, but the actual transfer performance degrades due to partial overlaps in the properties of the two robots. This can be captured by the variation of information metric.

Table 43 – Table showing the experimental results for one of the twenty trials. The "1-VI Metric" shows the results when the VI metric is used alone, while the "Estimate" column shows the results when the estimates from the methods in Section 5.8 are used. In combining the two, the best results are achieved. Here, both the "1-VI Metric" and "Estimate with VI" correctly pick the first robot as the best robot partner (bold) while the "Estimate" alone does not.

Partner Robot Number	Condition #	Properties Shared	1-VI Metric	Estimate	Estimate with VI	Transfer
1	1	1 2 3 4 5 6 7 8 10	**0.281**	0.747	**0.210**	**0.684**
2	2	1 2 3 4 7 8 9	0.175	0.770	0.123	0.640
3	3	1 2 3 4 5 6 7 8 9	0.246	0.727	0.179	0.669
4	4	1 4 5 6 7	0.112	0.735	0.082	0.588
5	5	1 2 3 4 5 6 7 8 9	0.234	0.727	0.170	0.632
6	6	1 3 4 5 6 8	0.242	0.723	0.175	0.614
7	7	1 2 3 4 5 6 8	0.248	**0.766**	0.190	0.663
8	8	1 2 3 5 6 7	0.095	0.699	0.066	0.614

Figure 77 plots these results in graphical form. Note that the "Estimate with VI" is not normalized to correspond to a true performance estimate. Since all that is needed is a maximal value in order to choose the best partner (argmax), normalization is not needed. Overall, there seems to be a strong correspondence to the estimate with VI compared to the *a priori* estimation alone. In other words, leveraging the VI metric seems to allow

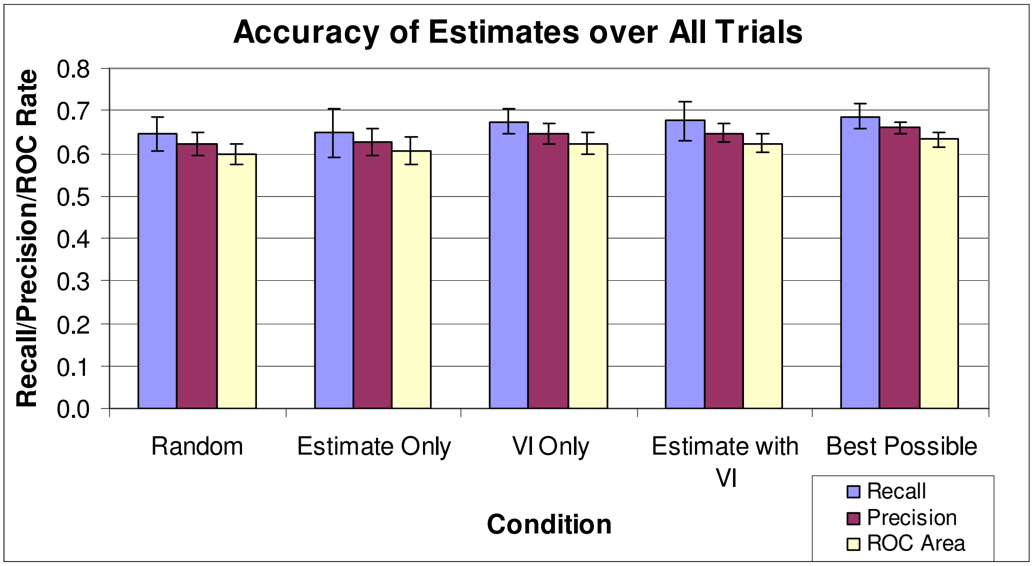


Figure 78 – Graph showing results averaged over all trials, for the three performance metrics. The "Estimate with VI" performs significantly better than randomly choosing a partner robot and also outperforms the "Estimate Only" control except for ROC area which was not significant. This shows that combining the performance estimates with the variation of information metric can provide significant boosts in post-transfer performance estimation. The last column shows the best possible performance if the best robot is chosen at each trial.

for better estimation of post-transfer performance; we now quantitatively verify this by showing results averaged over all trials.

Figure 78 shows the results for the five conditions (an additional "VI Only" condition was added) averaged over the twenty trials. Using the additional information provided by the variation of information metric performed both better than random (recall rate of 0.68, precision of 0.65, and ROC area of 0.62 compared to randomly choosing the partnerrobot which achieved 0.65 recall rate, 0.62 precision rate, and 0.60 ROC area, with all results being significant with p = 0.0013, precision p=0.0035 and ROC area p=0.0031). The difference between "Estimate with VI" (recall rate of 0.68, precision of 0.65, and ROC area of 0.62) and "Estimate" only (recall of 0.65, precision of 0.63, and

ROC area of 0.61) was also significant except for ROC area (p=0.0035 for recall, p=0.023 for precision, p=0.0791 for ROC area).

Note that the last condition on the right shows the best possible performance when the best robot is always chosen. In this case, the difference between the best and a randomly-chosen robot is significant (p=0.0001 for all metrics) but not great (a difference of about 5%) since there is redundancy in the properties and in many cases the missing properties did not affect performance greatly. In other words, the eight partner robots to choose from did perform differently, but in the future there should be greater differences to compare the methods. Future experiments should be run to reproduce these experiments where there is less redundancy between properties.

Again, these results show that the robot was able to leverage the information provided by the VI metric to better estimate the post-transfer performance of the receiving robot, with significant differences of about three to four percent. In other words, the correlation between the variation of information and knowledge transfer shown in 6.5 has proven to be useful in decision-making.

7.5. Summary

In this chapter, we have shown that the ability of robots to model their similarities and differences, using methods in chapters 4 and 6, is useful for more than just knowledge transfer. All of these experiments used real-robot configuration 2, with real-world data and objects. Specifically, we have demonstrated three different capabilities:

1) *Choosing a property to distinguish a concept from a set of concepts (Section 7.2)*. We hypothesized that the transferring robot would be able to use mappings between properties on the two robots (obtained via confusion matrices, as described in Section 4.2.1) and set operators to find a distinguishing property that would allow the receiving robot to pick out the target concept. We confirmed this hypothesis by running 100 trials where a target concept had to be distinguished from three concepts by the receiving robot. When the algorithm in Table 36 was used to determine the distinguishing property, the resulting accuracy of the receiving robot correctly picking the target concept was significantly higher than picking a concept at random or any other single property.

2) *Choosing a concept from a set of concepts that is estimated to be classified the best by the receiving robot (Section 7.4)*. Here, we combined the variation of information metric for shared properties between the robots (Section 6.3, algorithm in Table 29), the importance of the properties in the concept, and the *a priori* estimates (as described in Section 5.8) in order to pick the best concept from a set of eight concepts. Over a hundred randomized trials, the expert robot was able to use the algorithm in Table 38 to pick concepts that were classified significantly better by the receiving robot than picking a concept at random.

3) *Choosing the most similar robot in order to maximize performance after knowledge transfer (Section 7.4)*. Here, we combined the variation of information metric for shared properties between the robots (Section 6.3, algorithm in Table 29) and the *a priori* estimates (as described in Section 5.8) in order to choose a robot from a set of eight robots in order to maximize the post-transfer performance. We hypothesized that the variation of information metric could be leveraged by the expert robot to improve its

knowledge transfer estimate, and we confirmed the hypothesis. Using the algorithm in Table 41, the expert robot was able to pick a robot in order to obtain classification rates higher than a randomly-chosen robot as well as when using *a priori* estimates alone (without the VI metric).

In all, these results have combined the methods for calculating property mappings between robots (Section 4.2), the variation of information metric between property pairs of the two robots (Section 6.2), and *a priori* estimates of knowledge transfer (Section 5.8) to perform useful communication and decision-making. These capabilities are in addition to the knowledge transfer capabilities previously shown in Chapter 5 (e.g. Section 5.7).

CHAPTER 8

CONCLUSIONS AND FUTURE WORK

In this dissertation, we have introduced a framework for modeling and reasoning about perceptual differences between two heterogeneous robots, and using the resulting models for knowledge sharing and communication. In this final chapter, we will review the contributions made by the dissertation, discuss how they relate to the research questions laid out in Chapter 1, and plan a path forward in terms of future research directions.

8.1. Contributions

There are several contributions that have resulted from this dissertation. They include:

- **Demonstration that perceptual heterogeneity does indeed pose a problem for the transfer of learned object representations, even using modern computer vision algorithms that use object features specifically designed to be repeatable across images (Section 3.1).** We have conducted an experiment using three robots with different cameras, twelve real-world objects, and a state of the art computer vision algorithm to explore the importance of learning with the robot's own embodiment and the effect of perceptual differences on knowledge transfer. We showed that even when using features which are explicitly designed to be both repeatable and distinctive to particular objects, the highest accuracy is achieved when the robots use their own particular sensing to learn. Transfer from other robots can bootstrap learning, but can also result in catastrophic failures where the accuracy drops dramatically for certain objects due to missing features. This experiment demonstrated that, even in the best case scenario, perceptual heterogeneity can pose problems for knowledge transfer and that understanding the differences between the robots is important. Furthermore, we have shown that heterogeneity prevent properties, represented as Gaussian Mixture Models, from being transferred as well (Section 3.9).

- **Demonstration that using abstractions of lower-level sensory data to learn can facilitate not only learning itself but also knowledge transfer, compared to using the raw sensory data to learn.** We have conducted experiments using two robots with different sensors, thirty-four real-world objects, and a state of the art classification algorithm (support vector machines) analyzing whether our method for sensory abstraction actually improves learning and/or knowledge transfer. We demonstrated that it does improve both learning (Section 3.10) and knowledge transfer (Section 5.6), especially when the underlying sensory data used by the robots differ (e.g. one robot uses an RGB color space while another uses an HSV color space).

- **Algorithms and methods for implementing conceptual spaces and demonstration that they can be used to classify concepts using real-world noisy data.** Expanding the work of (Rickard, 2006), we developed a grounded representation of properties (e.g. 'green') and their combination for concepts (physical objects, e.g. 'apple') in Section 3.3, and developed algorithms for learning them from real data in Section 3.5 (algorithms in Table 5 for properties and Table 6 for concepts). We have also developed methods for learning these concepts even with missing data (Section 3.5) as well as adapting the resulting concepts when transferring them from one robot to another when some properties are not shared (Section 5.4).

- **Algorithms and representations suitable for learning models of perceptual differences between two robots at multiple levels, utilizing sensory data obtained after the two robots achieve a shared context.** We have demonstrated that, given instances from a shared context whereby robots are viewing the same scene, two robots can accurately build mappings between their respective properties (Chapter 4, experiments in Section 4.5). This was demonstrated using two different real-robot pairs (Section 4.5.2 and 4.5.3) as well as in simulation (Section 4.5.4)). In addition, we took potentially shared properties and further calculated how much information is lost when converting a concept from one robot's representation to another robot's representation, using information-theoretic measures (Chapter 6). The resulting models represent which properties and concepts are and are not shared by the two robots.

- **Methods for the *a priori* estimation of knowledge transfer performance.** We have shown that a robot can successfully estimate *a priori* how well a set of concepts (on average) will transfer by using its own performance using only shared properties (Section 5.8). In addition, we have applied an information theoretic metric (variation of information) in order to estimate information loss. After developing an algorithm for learning the metric from observations (Section 6.3), we showed that the metric does indeed measure the amount of overlap between properties (Section 6.4) and correlates inversely with knowledge transfer performance (Section 6.5).

- **Protocols and algorithms for using these models for knowledge exchange in several scenarios: transfer of a concept unknown to one robot, choice of properties that will distinguish an object in the receiving robot's representation, and choice of one concept over another based on whether information will be lost by the receiving robot.** We show how the models of similarities and differences between the robots can be used to perform these types of communication and knowledge transfer. We show how the models of similarities and differences between robots can be used to adapt existing knowledge by modifying the concepts to reflect missing properties (Section Section 5.4). Further, it is determined whether sufficient information is left to represent the concepts accurately (Sections 5.8 and 6.5). We have also demonstrated the ability to pick distinguishing properties to pick out one concept from a set of concepts, taking into account the robot's differences (Section 7.2). For example, these are useful for search and rescue domains where one robot must describe an object's appearance to another. We also showed similar capabilities for choosing a concept from a set of concepts such that the receiving robot would be able to classify it better than a randomly-chosen concept (Section 7.3).

- **Methods for choosing the best robot from a set of robots in order to maximize knowledge transfer efficacy.** By combining the *a priori* estimate (Section 5.8) as well as the variation of information metric (Chapter 6), we demonstrated that one robot can pick the best robot to transfer knowledge to from a set of robots, resulting in better post-transfer performance than picking a robot randomly (Section 7.4).

8.2. Research Questions Revisited

How can robots model their differences in perception to improve their ability to communicate, and how can the establishment of a shared context help, if at all?

We have demonstrated that robots can model their similarities and differences (Section 4.2) within a grounded multi-level representation (Section 3.2). This is possible by abstracting raw sensory data into an intermediate representation (properties, Section 3.3). A mapping between properties on each respective robot can then be built using instances from a shared context (Chapter 3). Furthermore, the variation of information metric, which leverages information theory, can be used to further model differences in partially overlapping properties (Chapter 6).

What is the role of abstraction of sensory data in communication and knowledge transfer?

The abstraction of raw sensory data was shown to be highly useful for both learning (Section 3.10) as well as knowledge transfer (Section 5.6). The abstractions represented higher-level characteristics of objects (e.g., color, texture, shape, and size) and essentially provided a buffer against lower-level sensory differences between the robots. This was especially true when the robots used different spaces to represent the same physical characteristics (e.g. RGB versus HSV representations for color properties) (Section 5.6).

What dialogues and protocols can allow two heterogeneous robots to use these models to align their knowledge and synchronize their symbols? How does the type of knowledge transfer possibly differ depending on the level of similarity that exists between the two robots?

We have developed several protocols for the building of similarity models (Section 4.4), knowledge transfer (Section 5.2), and communication between heterogeneous robots (Chapter 7). These methods allowed the robots to build mappings between their respective properties (Chapter 3) as well as information-theoretic models of differences in their properties (Chapter 6). As discussed in Section 5.1, different types of knowledge transfer are possible depending on what properties are shared, how they relate to the concepts, and how much they overlap.

How can these models be used to make the knowledge-sharing and communication processes sensitive to the capability differences between the robots?

The models of similarities and differences between the robots allowed the robots to not only transfer knowledge (Section 5.7), but also to estimate the performance of transfer *a priori* (Section 5.8), to pick distinguishing properties of a concept to differentiate it from its surrounding context (Section 7.2), to pick a concept from a set of concepts to maximize performance (Section 7.3), and to pick a robot from a set of robots to maximize knowledge transfer efficacy (Section 7.4). Overall, the explicit modeling of similarities and differences between the robots enabled these capabilities to outperform their controls.

How can these models be used to pick peer robots that are more similar in terms of properties and concepts, for a particular domain of knowledge?

By combining our methods for estimating knowledge transfer performance *a priori* (Section 5.8) with the information-theoretic metric for property overlap (Section 6.5), we were able to show that a robot can successfully choose one robot from a set in order to maximize knowledge transfer efficacy over a set of concepts (Section 7.4).

Primary Research Question

What interaction, dialogue, and adaptation processes are necessary to allow heterogeneous robots to model their differences, to use these models to exchange concepts and learning experiences, and how can such a system be used to improve the performance of a robot?

We have developed several protocols for the building of similarity models (Section 4.4), knowledge transfer (Section 5.2), and communication between heterogeneous robots (Sections 7.2, 0, and 7.4). These methods allowed the robots to build mappings between their respective properties (Section 4.5) as well as information-theoretic models of differences in their properties (Section 6.4 and 6.5). Performance was shown to improve after knowledge transfer both in terms of immediate classification accuracy after transfer (before the receiving robot has seen *any* training instances itself), as well as after the receiving robot continued learning in the form of higher learning curves (Sections 5.6 and 5.7). Furthermore, the chapter demonstrating various communication acts (Chapter 7) showed that even without knowledge transfer, the performance of a robot in picking out a target concept from its surroundings (Section 7.2) or in classifying a concept received from another robot (Section 7.3) is significantly better than the controls when using the methods outlined in the dissertation.

8.3. Future Work

We now discuss future research directions that stem from the research questions and framework presented in this dissertation.

8.3.1. Increasing the Feature Space

These days, there is an increasing availability of sensors that robots can have. For example, the now-familiar SICK ladar sensor has been expanded to allow three dimensional point clouds instead of being restricted to a plane. As a result, many more features or object properties can be sensed. This includes three dimensional shape characteristics, the sound objects make, the feel of an object given a touch sensor, or even chemical characteristics. It would be interesting to demonstrate that these additional object properties can be represented in the same conceptual spaces representation. These properties can be fused with the ones already used in this dissertation in order to increase the classification rates of the objects.

Furthermore, objects are more than their appearance. They can be felt, moved, pushed, dropped, sat on, etc. In other words, the affordance that an object provides is important for representing them (Gibson, 1977). Each affordance can potentially create a new space or domain with which to represent the object. For example, a "sittable" affordance can be represented using a continuous one-dimensional space. Properties in this space can carve out different degrees of applicability to the affordance. These types of properties can be combined with the normal perceptual properties in order to represent the concepts.

8.3.2. Social Symbol Grounding and Language

The grounding of symbols across a large group of agents is an open problem that is related to the work in this dissertation. Conceptual spaces has been proposed as a psychologically-inspired mechanism for bridging lower level sensory data to higher level symbols (Gärdenfors, 2000). We have made use of this fact in the form of properties, which provided a common abstraction of low-level sensory data between heterogeneous robots. It would be interesting to continue to explore these links by having robots that continually learn and transfer knowledge throughout their lives (i.e. life-long learning (Thrun & Mitchell, 1995)). Such an exploration could yield fruitful insight into how language and symbols can evolve through time.

Conceptual spaces also accounts for various characteristics of language, such as the modification of meaning depending on the context (e.g. “white” wine is different than a “white” wall). Future work can explore such capabilities using real robots that ground such symbols to sensors. Furthermore, the connection between such symbols can form ontologies, where concepts can be categorized into higher and higher levels. For example, an animal can be categorized based on the fact that it is a living thing, the type of animal it is, whether it is a mammal, etc. The building of such ontologies by robots using conceptual spaces is an interesting future direction, and investigation into the transfer of entire ontologies between robots becomes possible.

8.3.3. Applications to Transfer of Knowledge Relating to Action

Finally, this dissertation has specifically focused on perceptual heterogeneity. However, the problem of transferring *task* knowledge is still an open problem. Perception is an important aspect of knowing how to act. As a result, the transfer of perceptual knowledge can itself be useful for transferring task knowledge. For example, using the methods in this dissertation, two robots can transfer concepts that serve as indexes to a case-based reasoning system. After aligning the underlying indexes, the transfer of case knowledge becomes possible. Similarly, solutions to Markov Decision Process problems in the form of policies use the notion of an observable state. On a real robot, such states would consist of features that robots can perceive. Given a mapping between the perceptual states of two robots, it may be possible to transfer the policies themselves. Estimates of knowledge transfer, as developed in this dissertation (Sections

5.8 and 7.4) could serve useful for determining when the transfer of policies would be effective.

Another avenue of study could be the direct understanding and modeling of motor heterogeneity between two robots. This topic has been touched upon in the past (e.g. (Alissandrakis et al., 2002) but there is certainly room for further research. For example, it may be possible to abstract motor primitives, similar to what we have done with perceptual properties, in order to bridge motor heterogeneity between two robots. Such abstraction could allow robots to explicitly model their motor differences and determine when transfer is not possible if the differences are too large.

8.4. Final Words

The future of robotics is certain to eventually lead to collections of robots that interact in their environment. This dissertation has proposed the explicit modeling of difference between robots at a conceptual level in order to make such interactions effective. Several application areas can benefit from the deployment of the framework laid out here, namely tasks such as reconnaissance or search and research. The methods proposed in this dissertation are also not necessarily tied to mobile robots only, but can also be applied to distributed sensor networks containing heterogeneous sensors. This greatly extends the applicability of the work. As stated in the introduction, heterogeneity poses both challenges as well as opportunities. The framework laid out in this dissertation seeks to understand and overcome the challenges posed by heterogeneity, so it can be leveraged when it is needed. Furthermore, it is anticipated that leveraging theoretical fields such as information theory will continue to be useful in analyzing the output of sensory data, both within and across robots.

APPENDIX A

GLOSSARY OF TERMS

Attribute: A dimension of data used for classification. In this thesis, this can be a feature (which can be Boolean, nominal, or continuous) *or* a symbol (which is strictly Boolean).

Common Ground: Mutual knowledge, beliefs, and assumptions (Clark and Brennan, 1991) providing a foundation for knowledge exchange and communication. In humans, common ground is constantly updated through discourse (Clark and Brennan, 1991), and estimation of it can be done via social cues, experience, etc. (Kiesler, 2005).

Concept: A combination of regions in a set of domains, along with salience weights and correlations between its properties.

Confusion Matrix: A matrix representing co-occurrence of items in rows and columns, usually for gauging accuracy of a classifier. In our case, we use it to map properties and concepts in two different robots based on their co-occurrence given data for the same property or concept from each.

Context:

- General Definition: The interrelated conditions in which something exists or occurs (Merriam-Webster.com, 2008a)
- Working Definition: We use the notion of context in a few ways, and separate them into *local context* (of which *physical context* is a specialization) and *global context.* In general, context defines specific constraints on a situation, whether it be perceptual (e.g. the distance to a target object) or symbolic (e.g. task constraints or robot state). See also **Locally-Shared Context** and **Globally-Shared Context**.

Context Function: A context function is a function that takes in perceptual data and returns 1 if all of the constraints on the data defined by the context are met.

Context Method: A method that can return sensory data with all of the constraints of a context function satisfied, either by physically moving to an appropriate location in the environment or obtaining suitable sensory readings from memory.

Difference Models: Models of shared domains, properties, and concepts that measure similarity between corresponding pairs from each robot. (This is synonymously used with "Similarity Models", as the model can be used to ascertain what is different as well).

Dimension: A dimension or axis in a domain, representing a physical characteristic (e.g. pitch for sound).

Domain: A geometric space consisting of a set of integral dimension.

Domain Similarity Model: Model of similarity between domains (usually of two different robots), describing whether they measure similar aspects of the world. In our case, we infer this based on similarities in higher-level properties.

Expert Robot: In our knowledge transfer experiments, the expert robot is the one that learns, using its own instances and sensors, representations for a set of concepts. This robot then sends its representations to the non-expert or receiving robot.

Feature: See **perceptual feature.**

Feature Value: Specific value of a perceptual feature at an instance in time.

Globally Shared Context: A context that places constraints on perceptual features or symbols that may or may not be grounded in perception (examples of ungrounded symbols include task or state symbols). A globally *shared* context is where two robots share this context, i.e. both produce data that meet the same (or related) constraints.

Gaussian Mixture Model (GMM): A probabilistic model of a density that consists of a weighted combination of multiple Gaussians.

Heterogeneity: See **Perceptual Heterogeneity**. Other types of heterogeneity, such as motor heterogeneity, are not dealt with in this thesis.

Integral Dimension: Dimensions in a conceptual space that cannot be separated, usually representing similar physical properties.

Learning Curve: A learning curve is a curve plotting a performance metric on the y-axis (e.g. recall) as the number of training instances increase (along the x-axis).

Locally Shared Context: A context that places constraints on only perceptual features or symbols that are directly grounded in immediate perception. A locally *shared* context is where two robots share this context, i.e. both produce data that meet the same (or related) constraints.

Object: A specific physical entity that can be represented, in our case, using a knoxel in a conceptual space.

Object Category: An object category defines higher-level characteristics of objects that share important properties, but may vary in specifics such as location, size, color, etc. This distinction is difficult to learn in general, and in this work utilizes supervised learning to map different instances of an object category to the same label, while also providing specific labels for object category instances as well.

Observation (Vector): A vector of data obtained at an instant of time from a sensor.

Ontology: A hierarchical description of concepts, in our cases mostly physical objects or object categories. The specific representation we use are conceptual spaces, which define concepts based on a set of regions in a set of domains. It is hierarchical (or structured) in the sense that they can be nested or combined in various ways, for example the concept 'navy blue' can be a subset of the concept 'blue'. Properties can be tied to symbols as well.

Ontology Similarity Model: Model of similarity between symbols (usually of two different robots) tied to concepts.

Peer Robots: Other robots with whom a particular robot will have to communicate or exchange knowledge with to perform a task.

Perceptual Feature: A set of vector values corresponding to some form of processing performed on sensor or other perceptual feature data in salient parts of the sensory readings. Examples include color segments, histograms, or lines obtained from a camera sensor.

Perceptual Heterogeneity: In the English language, Heterogeneity is defined as the quality or state of being heterogeneous, which is defined as consisting of dissimilar or diverse ingredients or constituents (Merriam-Webster.com, 2008b). Perceptual heterogeneity refers to consisting of differing sensors or perceptual features (which process the sensors). In our work, we describe several categories corresponding to different levels at which the robots can differ (namely, at the domain, property, and concept levels).

Perceptual Shared Context: See **Locally Shared Context**.

Physically Shared Context: A context that places constraints on only perceptual features or symbols that are directly grounded in immediate perception, and more specifically on sensors and perceptual features corresponding to physical location in an environment. A physically *shared* context is where two robots share this context, i.e. both produce data that meet the same (or related) constraints.

Precision: The precision of a binary classifier is the number of true positives divided by the sum of true positives and false positives.

Property: A region in one domain, in our case represented by a Gaussian Mixture Model clustering. For example, 'blue' can be a region in an HSV color space.

Property Mapping Model: Model of similarity between properties (usually of two different robots), describing whether they measure similar aspects of the world. The model in this work is in the form of a confusion matrix, and the actual mapping can be obtained by taking maximal values for each row, optionally thresholded. This results in properties on robot A that map to properties on robot B.

Property Similarity Model: Model of similarity between properties (usually of two different robots), describing whether they measure similar aspects of the world. The model in this work is in the form of a confusion matrix as well as calculated values for loss of information when going from a property in one robot to a property in a different robot (see **Variation of Information**).

Recall: The recall of a binary classifier is the number of true positives divided by the sum of true positives and false negatives.

Receiving Robot: In our knowledge transfer experiments, the receiving (or non-expert) robot is the one receives learned representations from the expert robot.

ROC Area: Area under the **ROC Curve**.

ROC Curve: Receiver operating characteristic (ROC) curve, which plots the true positive ratio against the false positive ratio. The true positive ratio, or sensitivity, of a binary classifier is the number of true positives divided by the total number of positives in the test set. The false positive ratio is the number of false positives divided by the total number of negatives in the test set.

Sensor: A device that responds to a physical stimulus (as heat, light, sound, pressure, magnetism, or a particular motion) and transmits a resulting impulse (as for measurement or operating a control) (Merriam-Webster.com, 2008c).

Sensory Data: Data from either 1) sensors or 2) perceptual features that process data from sensors.

Shared Context: A context that is shared between two robots, i.e. both produce data that meet the same (or related) constraints. We specialize this in the form of a **globally shared context**, **locally shared context**, and **physically shared context**.

Similarity Models: Models of shared domains, properties, and concepts that measure similarity between corresponding pairs from each robot. (This is synonymously used with "Difference Models", as the model can be used to ascertain what is different as well)

Structured Knowledge Sharing: Knowledge sharing that results in the sharing of the representational structure corresponding to a symbol. An example is the concept sharing algorithm in Section 5.2, given that it is successful.

Support Vector Machines: A discriminative classification method. We used the svm^{light} library that implements the algorithm (Joachims, 1999).

Symbol: A label corresponding to a specific representation or grounding, in our case a concept in a conceptual space, representing multiple regions in a set of domains. The label is the *referent* and the representation is the *form* in the semiotic triangle (Vogt, 2007).

Symbolic Shared Context: See **Globally Shared Context**

Variation of Information (VI): An information-theoretic metric that measures how much information is lost when using one clustering over another. Since a property is a region in a domain, and we represent these regions as a clustering, this metric can be used to quantify information loss when describing a property in one robot versus another.

APPENDIX B

SUMMARY OF NOTATION

Sensors

Set of sensors: $S = \{s_1, s_2, ..., s_m\}$

Number of sensors: $|S|$

Set of observations at time *t*: $O_t = \{o_{t,1}, o_{t,2}, ..., o_{t,|S|}\}$

Sensor *i* of robot *A*: s_i^A

Features

Set of perceptual features: $F = \{f_1, f_2, ..., f_p\}$ where $f_i = \Phi(\mathbf{O}_{f_i})$

where $\mathbf{O}_{f_i} \subseteq \mathbf{O}$ denotes the set of input observations used by the feature detector

Feature set of robot *A*: F^A

Specific values of a set of features at time *t*: $\mathbf{F}_t$

Specific value of a feature *i*: $\mathrm{f}_{t,i}$

Conceptual Space Representations

Domain: $D = \{d_1, d_2, ..., d_n\}$

Knoxel in conceptual space: $k = < k_1, k_2, ..., k_n >$

Symbol Set: X

Representation of symbol $x \in \mathrm{X}$, knoxel k: $R : (k \ \ , x) \rightarrow [0,1]$

Property set: $P \subseteq \mathrm{X}$

Prototype for property $p \in P$: k_p

Weight between property *i* and *j* in connection matrix C of a concept: $C_{(i,j)}$

Concept vector: c

Similarity between concept *c* and *c'*: $s(c, c')$

Similarity between concept *c* and instance *i*: $s(c, i)$

Domains in conceptual space for concept *c*: D_c

Set of all properties involved in concept *c*: P_c

(i.e. $P_c = \{ p \in P : \exists j \text{ s.t.} C_{pj} > 0 \text{ or } C_{jp} > 0 \}$)

Instance matrix for concept *c*: $I_{c,(i,j)}$

Set of knoxels from perceptual feature detectors: K

Knoxel for instance *i*: k_i

Gaussian Mixture Model for property *j*: G_j

Property confusion matrix: $PC_{(j,k)}$

Concept confusion matrix: $CC_{(j,k)}$

Mutual Information (VI) for two clusterings *G* and *G'*: $I(G_i^A, G_j^B)$

Variation of Information (VI) for two clusterings *G* and *G'*: $VI(G_i^A, G_j^B)$

Other

List of peer robots: P^A

REFERENCES

Aamodt, A. & Plaza, E. (1994), 'Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches', *Artificial Intelligence Communications* **7**(1), pp. 39-52.

Aisbett, J. & Gibbon, G. (2001), 'A general formulation of conceptual spaces as a meso level representation', *Artificial Intelligence* **133**(1-2), pp. 189-232.

Alissandrakis, A.; Nehaniv, C.L. & Dautenhahn, K. (2002), 'Do as I Do: Correspondences across Different Robotic Embodiments', *in* 'Proceedings of the 5th German Workshop on Artificial Life'.

Alisssandrakis, A.; C. L. Nehaniv & Dautenhahn, K. (2003), 'Solving the correspondence problem between dissimilarly embodied robotic arms using the ALICE imitation mechanism', *in* 'Proceedings of the Second International Symposium on Imitation in Animals & Artifacts', pp. 79-92.

Arkin, R.C. (1997), 'AuRA: principles and practice in review', *Journal of Experimental & Theoretical Artificial Intelligence* **9**(2), pp. 175-189.

Baillie, J. (2004), 'Grounding Symbols in Perception with two Interacting Autonomous Robots', *in* 'Proceedings of the Fourth International Workshop on Epigenetic Robotics: Modeling Cognitive Development in Robotic Systems', pp. 107-110.

Balakirsky, S.; Scrapper, C.; Carpin, S. & Lewis, M. (2006), 'UsarSim: providing a framework for multirobot performance evaluation', *in* 'Proceedings of PerMIS', pp. 98--102.

Balch, T. (2005), *Communication, Diversity and Learning: Cornerstones of Swarm Behavior*, Springer Verlag.

Balch, T. (1998), 'Behavioral diversity in learning robot teams', PhD thesis, College of Computing, Georgia Institute of Technology.

Balkenius, C.; Gärdenfors, P. & Hall, L. (2000), 'The Origin of Symbols in the Brain', in 'Proceedings of the Conference on the Evolution of Language', Ecole Nationale Superieure des Telecommunications, pp. 13-17.

Bauer, E. & Kohavi, R. (1999), 'An empirical comparison of voting classification algorithms: Bagging, boosting, and variants', **36**(105).

Becker, R. & Corkill, D. (2007), 'Determining Confidence when Integrating Contributions from Multiple Agents', *in* 'Proceedings of the Sixth International Joint Conference on Autonomous Agents and Multi-Agent Systems'.

Billard, A. & Dautenhahn, K. (1999), 'Experiments in learning by imitation - grounding and use of communication in robotic agents', *Adaptive Behavior* **7**(3/4), pp. 415-438.

Bilmes, J. (1998), 'A Gentle Tutorial of the EM Algorithm and its Application to Parameter Estimation for Gaussian Mixture and Hidden Markov Models', *International Computer Science Institute* **4**.

Britanik, J. & Marefat, M. (1993), 'Distributed case-based planning: Multi-agent cooperation for high autonomy', *in* 'AI, Simulation, and Planning in High Autonomy Systems: Integrating Virtual Reality and Model-Based Environments', pp. 295-301.

Cangelosi, A. (2001), 'Evolution of communication and language using signals, symbols, and words', *IEEE Transactions on Evolutionary Computation* **5**(2), pp. 93--101.

Carpin, S., Wang, J., Lewis, M., Birk, A., & Jacoff, A. (2005). 'High Fidelity Tools for Rescue Robotics: Results and Perspectives', *in* 'Proceedings of RoboCup 2005', Osaka, Japan.

Chella, S.C.; Frixione, M. & Saffiotti, A. (2004), 'Perceptual Anchoring via Conceptual Spaces', *in* 'Proc. of the AAAI-04 Workshop on Anchoring Symbols to Sensor Data', AAAI Press.

Clark, H. & Wilkes-Gibbs, D. (1986), 'Referring as a Collaborative Process', *Cognition* **22**(1), pp. 1-39.

Clark, H.H. & Brennan, S.A. (1991), *Grounding in communication*, Washington: APA Books, chapter 7, pp. 127-149.

Coradeschi, S. & Saffiotti, A. (2000), 'Anchoring Symbols to Sensor Data: preliminary report', *in* 'Proc. of the 17th AAAI Conf', pp. 129-135.

cs.unc.edu (2008). *Open source computer vision library: Reference guide*. Retrieved February 13, 2008 from http://www.cs.unc.edu/Research/stc/FAQs/OpenCV/OpenCVReferenceManual.pdf.

Dautenhahn, K. & Nehaniv, C.L. (2002), '*The Correspondence Problem, Imitation in Animals and Artifacts*', *in Imitation in Animals and Artifacts*, MIT Press, chapter 2.

Deacon, T. (1997), *The Symbolic Species: The co-evolution of language and the human brain*, Penguin Books.

van Diggelen, J.; Beun, R.; Dignum, F.; van Eijk, R. & Meyer, J. (2005), 'A decentralized approach for establishing a shared communication vocabulary', *in* 'International workshop on agent mediated knowledge management', Ultrecht University, The Netherlands.

Donald, M. (1991), *Origins of the Modern Mind: Three Stages in the Evolution of Culture and Cognition*, Harvard University Press.

Drescher, G. (1991), *Made-up Minds: A Constructivist Approach to Artificial Intelligence*, MIT Press.

Ehrig, M. & Euzenat, J. (2004), 'State of the art on ontology alignment', *Knowledge Web Deliverable* **2**(3).

Felzenszwalb, P.F. and Huttenlocher, D.P., ‘Efficient Graph-Based Image Segmentation’, *International Journal of Computer Vision*, **59**(2), pp. 167-181.

Fong, T.; Nourbakhsh, I. & Dautenhahn, K. (2003), 'A survey of socially interactive robots', *Robotics and Autonomous Systems* **42**, pp. 143-166.

Fowlkes, E. & Mallows, C. (1983), 'A method for comparing two hierarchical clusterings', *Journal of the American Statistical Association* **78**(383), pp. 553-569.

Gärdenfors, P. (2000), *Conceptual Spaces: The Geometry of Thought*, MIT Press.

Gardenfors, P. & Williams, M. (2001), 'Reasoning about categories in conceptual spaces', *in* 'Proceedings of the Fourteenth International Joint Conference of Artificial Intelligence', pp. 385-92.

Gerkey, B. & Mataric, M. (2004), 'A formal analysis and taxonomy of task allocation in multi-robot systems', *International Journal of Robotics Research* **23**(9), pp. 939-954.

Gerkey, B.; Vaughan, R. & Howard, A. (2003), ''The Player/Stage Project: Tools for Multi-Robot and Distributed Sensor Systems, *in* 'Proceedings of the 11th International Conference on Advanced Robotics', pp. 317--323.

Gibson, J.J. (1977). The theory of affordances. In R. Shaw & J. Bransford (eds.), *Perceiving, Acting and Knowing*. Hillsdale, NJ: Erlbaum.

Harnad, S. (1990), 'The Symbol Grounding Problem', *Physica D* **42**, pp.335-346.

Hayes, C.; Avesani, P. & Cova, M. (2005), 'Language Games: Learning Shared Concepts among Distributed Information Agents', *in* 'IJCAI-05 Workshop: Multi-Agent Information Retrieval and Recommender Systems'.

Intel.com (2008), *Open CV Library Overview – Open Source Computer Vision Library*. Retrieved on February 13, 2008 from http://www.intel.com/technology/computing/opencv/overview.htm.

Ionescu, A. (2006), 'Extension of the Concept Representation in Conceptual Spaces', *in* 'Annual meeting of the North American Fuzzy Information Processing Society', pp. 296-301.

Isaacs, E.A. & Clark, H.H. (1987), 'References in conversations between experts and novices', *Journal of Experimental Psychology: General* **116**, pp.26-37.

Isukapalli, R.; Elgammal, A. & Greiner, R. (2006), 'Learning to Detect Objects of Many Classes Using Binary Classifiers', *in* 'Proceedings of the 9th European Conference on Computer Vision', pp. 352-364.

Joachims, T., 'Making large-Scale SVM Learning Practical'. *Advances in Kernel Methods - Support Vector Learning*, B. Schölkopf and C. Burges and A. Smola (ed.), MIT-Press, 1999.

Jung, D. & Zelinsky, A. (2000), 'Grounded Symbolic Communication between Heterogeneous Cooperating Robots', *Auton. Robots* **8**(3), pp. 269--292.

Kaplan, V. (2006), 'The challenges of joint attention', *Interaction Studies* **7**(2), pp. 135--169.

Khoo, A. & Horswill, I. (2003), 'Grounding Inference in Distributed Multi-Robot Environments', *Journal of Robotics and Autonomous Systems, Special issue on perceptual anchoring* **43**(2-3), pp. 85-96.

Kiesler, S. (2005), 'Fostering Common Ground In Human-Robot Interaction', *in* 'IEEE International Workshop Robots and Human Interactive Communication', IEEE Press, pp. 729-734.

Kira, Z. & Long, K. (2007), 'Modeling Robot Differences by Leveraging a Physically Shared Context', *in* M. Littman H. Kozima & C. Balkenius L. Berthouze, C.G. Prince, ed.,'Proceedings of the International Conference on Epigenetic Robotics: Modeling Cognitive Development in Robotic Systems', pp. 53-59.

Kira, Z. (2009a), 'Mapping Grounded Object Properties across Perceptually Heterogeneous Embodiments', *in* 'Proceedings of the 22nd International FLAIRS Conference', pp. 57-62.

Kira, Z. (2009b), 'Transferring Embodied Concepts between Perceptually Heterogeneous Robots', *in* 'Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems', pp. 4650-4656.

Kira, Z. (2010), 'Inter-Robot Transfer Learning for Perceptual Categorization', accepted to the *9th International Conference on Autonomous Agents and Multiagent Systems* (AAMAS), Toronto, Canada.

Kirby, S. (2002), 'Natural Language From Artificial Life', *Artificial Life* **8**(2), pp. 185-215.

Kraus, S.; Nirkhe, M. & Sycara, K.P. (1993), 'Reaching agreements through argumentation: a logical model (Preliminary report) ', *in* 'Proceedings of the 12th International Workshop on Distributed Artificial Intelligence', pp. 233--247.

Kraut, F.S.R. & Siegel, J. (2003), 'Visual information as a conversational resource in collaborative physical tasks', *Human-Computer Interaction* **18**(1), pp. 13-49.

Laera, L.; Blacoe, I.; Tamma, V.; Payne, T.R.; Euzenat, J. & T., B. (2007), 'Argumentation over Ontology Correspondences in MAS', *in* 'Proceedings of Sixth International Joint Conference on Autonomous Agents and Multiagent Systems'.

Lander, S.E. & Lesser, V.R. (1992), 'Customizing Distributed Search Among Agents with Heterogeneous Knowledge', *in* 'Proceedings of the First International Conference on Information and Knowledge Management', pp. 335-344.

Leake, D. & Sooriamurthi, R. (2002), 'Managing multiple case-bases: Dimensions and issues', *in* 'Proceedings of the Fifteenth FLAIRS Conference', AAAI Press, pp. 106-110.

LeBlanc, K. & Saffiotti, A. (2007), 'Issues of Perceptual Anchoring in Ubiquitous Robotic Systems', *in* 'Proc. of the ICRA-07 Workshop on Omniscient Space'.

Lehmann, F. (1992), *Semantic Networks in Artificial Intelligence*, Elsevier Science Inc. New York, NY, USA.

Likhachev, M.; Kaess, M. & Arkin, R. (2002), 'Learning Behavioral Parameterization Using Spatio-Temporal Case-Based Reasoning', *in* 'Proceedings of the IEEE International Conference on Robotics and Automation', pp. 1282-1289.

Lowe, D. (1999), 'Object recognition from local scale-invariant features', in 'Proceedings of the International Conference on Computer Vision, pp. 1150-1157.

Lungarella, M.; Metta, G.; Pfeifer, R. & Sandini, G. (2003), 'Developmental robotics: a survey', *Connection Science* **15**(4), pp. 151-190.

MacKenzie, A.R. & Cameron, J. (1997), 'Multiagent Mission Specification and Execution', *Autonomous Robots* **4**(1), 29--52.

Matson, E. & DeLoach, S. (2003), 'Using Dynamic Capability Evaluation to Organize a Team of Cooperative, Autonomous Robots', *in* 'Proceedings of the International Conference on Artificial Intelligence', pp. 23--26.

McGinty, L. & Smyth, B. (2001), 'Collaborative Case-Based Reasoning: Applications in Personalised Route Planning', *in* 'Proceedings of the International Conference on Case-Based Reasoning', pp. 362-376.

Meila, M. (2002), 'Comparing Clusterings', Technical report 418, University of Washington.

Messina, E.; Jacoff, A.; Scholtz, J.; Schlenoff, C.; Huang, H.; Lytle, A. & Blitch, J. (2005), 'Statement of requirements for urban search and rescue robot performance standards, preliminary report'.

Merriam-Webster.com (2008a), *context – Definition from the Merriam-Webster Online Dictionary*. Retrieved February 12, 2008 from http://www.merriam-webster.com/dictionary/ontology.

Merriam-Webster.com (2008b), *heterogeneity – Definition from the Merriam-Webster Online Dictionary*. Retrieved February 12, 2008 from http://www.merriam-webster.com/dictionary/heterogeneity.

Merriam-Webster.com (2008c). *sensor – Definition from the Merriam-Webster Online Dictionary*. Retrieved February 12, 2008 from http://www.merriam-webster.com/dictionary/sensor.

Mitchell, T. (1997), *Machine Learning*, McGraw Hill.

Nagendra, M.; Lesser, V. & Lander, S. (1995), 'Retrieval and Reasoning in Distributed Case Bases', *UMass Computer Sciences Technical Report 95-27*.

Neches, R. & R. Fikes, T.G.R.P.T.S.&.W.R.S. (1991), 'Enabling technology for knowledge sharing', *AI Magazine* **12**(3), 16-36.

Nistér, D. and Stewénius, H., 'Scalable recognition with a vocabulary tree', *in* 'Proceedings of the IEEE Computer Society Conference on. Computer Vision and Pattern Recognition', volume 2, pp. 2161-2168.

Nottale, M. & Baillie, J. (2007), 'Talking Robots: Grounding a Shared Lexicon in an Unconstrained Environment', *in* 'Proceedings of the 7th International Conference on Epigenetic Robotics'.

Ontañón, S. (2005),'Ensemble Case Based Learning for Multi-Agent Systems', PhD thesis, Universitat AutrïSnoma De Barcelona.

Ontañón, S.; Mishra, K.; Sugandh, N. & Ram, A. (2007), 'Case-Based Planning and Execution for Real-Time Strategy Games', *in* 'Proceedings of the Seventh International Conference on Case-Based Reasoning', pp. 164-178.

Ontañón, S. & Plaza, E. (2006), 'Arguments and Counterexamples for Case-based Joint Deliberation', *in* 'Proceedings of the Third International Workshop on Argumentation in Multi-Agent Systems', pp. 36-53.

Ontañón, S. & Plaza, E. (2003), 'Collaborative Case Retention Strategies for CBR Agents', *in* 'Proceedings of the International Conference on Case Based Reasoning', pp. 392-406.

Ontañón, S. & Plaza, E. (2002), 'A bartering approach to improve multiagent learning', *in* 'Proceedings of the International Conference on Autonomous Agents and Multiagent Systems (AAMAS)', pp. 386-393.

Ontañón, S. & Plaza, E. (2001), 'Learning when to collaborate among learning agents', *Lecture Notes in Artificial Intelligence 2167*, Springer-Verlag, , pp. 394-405.

Parker, L. (2003), 'The effect of heterogeneity in teams of 100+ mobile robots', *Multi-Robot Systems*, Schultz, Alan C.; Parker, Lynne E.; Schneider, Frank E. (Eds.) pp. 205-229.

Parker, L. (2000), 'Lifelong Adaptation in Heterogeneous Multi-Robot Teams: Response to Continual Variation in Individual Robot Performance', *Autonomous Robots* **8**(3), pp. 239-267.

Prasad, M.; Lander, S. & Lesser, V. (1995), 'On retrieval and reasoning in distributed case bases', *in* 'Proceedings of the IEEE Conference on Systems, Man and Cybernetics', pp. 351-356.

Prasad, M.V.N. (2000), 'Distributed Case-Based Learning', *in* 'Proceedings of the Fourth International Conference on MultiAgent Systems', pp. 222-229.

Ram, A. & Santamaria, J.C. (1997), 'Continuous Case-Based Reasoning', *Artificial Intelligence* **90**(1-2), 25--77.

Ramsey, C.L. & Grefenstette, J.J. (1993), 'Case-Based Initialization of Genetic Algorithms', *in* 'Proceedings of the 5th International Conference on Genetic Algorithms', Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp. 84--91.

Rand, W. (1971), 'Objective criteria for the evaluation of clustering methods', *Journal of the American Statistical Association* **66**(336), pp. 846-850.

Raubal, M. (2004), 'Formalizing Conceptual Spaces', *in* 'Proceedings of the Third International Conference on Formal Ontology in Information Systems', pp. 153-164.

Rickard, J. (2006), 'A concept geometry for conceptual spaces', *Fuzzy Optimization and Decision Making* **5**(4), pp. 311--329.

Rosch, E. (2004), 'Principles of Categorization', in *Fuzzy Grammar: A Reader*, Oxford University Press, USA.

Rosch, E. (1988), *Readings in Cognitive Science*, Morgan Kaufmann, chapter Principles of Categorization, pp. 312-322.

Roy, D. (2005), 'Grounding words in perception and action: computational insights', *Trends Cogn Sci* **9**(8), pp. 389-396.

Roy, D. (2005), 'Semiotic schemas: A framework for grounding language in action and perception', *Artificial Intelligence* **167**(1), pp. 170-205.

Saunders, J., Nehaniv, C. L., & Dautenhahn, K. (2006). 'Teaching robots by moulding behavior and scaffolding the environment', *Human-Robot Interaction* , vol. 2006, pp. 118-125.

Scassellati, B. (2002), 'Theory of Mind for a Humanoid Robot', *Autonomous Robots* **12**(1), 13--24.

Steels, L. (2003), 'Evolving Grounded Communication for Robots', *Trends in Cognitive Sciences* **7**(7), 308-312.

Steels, L. & Kaplan, F. (1999), *Bootstrapping Grounded Word Semantics*, in Briscoe, T., editor, *Linguistic evolution through language acquisition: formal and computational models*, pp. 53-74, Cambridge University Press. Cambridge, UK.

Steels, L. & Loetzsch, M. (2007), *Spatial Language and Dialogue*, Oxford University Press, chapter Perspective Alignment in Spatial Language.

Steels, L. & Vogt, P. (1997), 'Grounding adaptive language games in robotic agents', *in* 'Proceedings of the 4th European Conference on Artificial Life', pp. 474-482.

Stoytchev, A. (2003), 'Computational Model for an Extendable Robot Body Schema', Technical report GIT-CC-03-44, Georgia Institute of Technology.

Stoytchev, A. (2009), 'Some Basic Principles of Developmental Robotics', *IEEE Transactions on Autonomous Mental Development*, **1**(2), pp. 122-130.

Stubbs, K.; Hinds, P.J. & Wettergreen, D. (2007), 'Autonomy and Common Ground in Human-Robot Interaction: A Field Study', *IEEE Intelligent Systems* **22**(2), pp. 42-50.

Swain, D. (1990), 'Indexing via color histograms', *in* 'Proceedings of the Third International Conference on Computer Vision', pp. 390-393.

Thrun, S. and Mitchell, T.M. (1995), 'Lifelong robot learning', *Robotics and autonomous systems* **15**(1), pp. 25-46.

Trafton, J.G.; Cassimatis, N.L.; Bugajska, M.D.; Brock, D.P.; Mintz, F.E. & Schultz, A.C. (2005), 'Enabling effective human-robot interaction using perspective-taking in robots', *IEEE Transactions on Systems, Man, and Cybernetics---Part A: Systems and Human*, **35**(4), pp. 460-470.

Thrun, S. & Liu, Y. (2005), 'Multi-robot SLAM with Sparse Extended Information Filers', *in* 'Proceedings of the 11th International Symposium of Robotics Research', pp. 254-266.

Ulam, P. & Balch, T. (2003), 'Niche Selection in Foraging Tasks in Multi-Robot Teams Using Reinforcement Learning', *Adaptive Behavior* **12**(4).

Ulam, P., Endo, Y., Wagner, A., and Arkin, R.C. (2007), 'Integrated Mission Specification and Task Allocation for Robot Teams - Design and Implementation', *in* 'Proceedings of IEEE International Conference on Robotics and Automation', pp. 4428-4435.

Vogt, P. (2003), 'Anchoring of semiotic symbols', *Robotics and Autonomous Systems* **43**(2), 109-120.

Vogt, P. & Divina, F. (2007), 'Social symbol grounding and language evolution', *Interaction Studies* **8**(1).

Watson, I. & Gardingen, D. (1999), 'A Distributed Case-Based Reasoning Application for Engineering Sales Support', *in* 'Proceedings 16th International Joint Conference on Artificial Intelligence', pp. 600-605.

Weiss, G. & Dillenbourg, P. (1999), 'What is 'multi' in multi-agent learning? ', *in* P. Dillenbourg, ed.,'Collaborative-learning: Cognitive and Computational Approaches', Oxford : Elsevier, pp. 64-80.

Wertsch, R. (1995), 'Vygotsky on learning and development', *Human Development* **38**, 332-37.

Yanco, H. & Stein, L.A. (1992), 'An adaptive communication protocol for cooperating mobile robots', *in* 'Proceedings of the second international conference on From animals to animats 2 : simulation of adaptive behavior: simulation of adaptive behavior', pp. 478-485.